1993

Disjunctively incomplete information in relational databases: modeling and related issues

Lu Zhang
Iowa State University

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

University Microfilms International
A Bell & Howell Information Company
300 North Zeeb Road. Ann Arbor. MI 48106-1346 USA
313/761-4700 800/521-0600

Order Number 9321231

Disjunctively incomplete information in relational databases: Modeling and related issues

Zhang, Lu, Ph.D.

Iowa State University, 1993

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

Disjunctively incomplete information in relational databases: Modeling and related issues

by

Lu Zhang

A Dissertation Submitted to the
Graduate Faculty in Partial Fullfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Department: Computer Science
Major: Computer Science

Approved:

Members of Committee:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames Iowa

1993

TABLE OF CONTENTS

1. INTRODUCTION

It has been noted in the literature (e.g., [Kers90], [MiWi86], [MyBr91], [PaZe91], and [Smit86]) that the next generation of computerized information systems must have not only the ability to store and to access but also to reason about a large volume of information and knowledge. This new breed of information systems centers on an exciting confluence of many technologies and must be developed using an interdisciplinary approach and further developments in the field of data modeling. In addition to being intelligent, it should also exhibit the characteristics of being distributed (e.g., being able to interface/integrate with a heterogeneous information processing environment consisting of various database management systems) and being objected-oriented. Generally speaking, the trend is to consolidate the information system technology with the advancements in other areas so that future information systems will become an inseparable component of the future information age and technology rather than a segregated island.

On the other hand, database systems themselves are evolving in the direction of incorporating and expressing more semantics and a wider range of real world information in the conceptual level.

In essence, the next generation information systems need further research. The focus of the research work explored in this dissertation is related to the data modeling aspect associated with this next generation of information systems. Specifically, the recurrent and prominent problem of modeling null values in the form of inclusively and exclusively disjunctive forms within the realm of the relational model is addressed. In other words, this thesis considers the proposition of incorporating incomplete information into the relational model and the problems raised by information incompleteness in the context of the relational model, or how to meaningfully interpret and process incomplete information. Comparatively speaking, the modeling approach

shifts the representation and management of uncertainty from the inference engine of the intelligent systems to the database and the database manipulation language.

The relational approach to database systems assumes that data is logically structured as relations. A relation can be viewed as a table, with each column in the table corresponding to an attribute of an entity while each row in the table represents an association among the attributes. The rows are called tuples of the relation. The relational model assumes that all the values of the tuples are specified. That is, there is no incomplete information. However, in practice it is often the case that the available information is incomplete, either due to human being's incomplete perception of the real world or due to the unavailability of the required information. Consider, for example, a talent database for a large corporation which stores information of employees who wish to be considered for jobs within the company. Suppose there is a relation within such system that keeps the name of such candidate employees along with the associated attributes such as age, sex, current position, current salary, etc. However, due to the practice of Affirmative Action, the employer is not supposed to discriminate based on ages and sexes in filling positions. Consequently, the sex and age information is optional. Therefore, for some candidate records, the age and/or sex columns are unknown and hence are left to be blanks or special null symbols.

The above motivating example illustrates one facet of the many ideographs of null values. Null values represent a form of uncertainty or vagueness which can be introduced by a variety of sources. Uncertainties are natural manifestations of information reliability. This type of uncertainty brings out, in many cases, the uncertainties that present in the factual knowledge due to, for instance, inaccuracy and poor reliability of knowledge gathering mechanisms, judgemental decisions, subjective preferences among different possibilities, ill-defined observations, faithlessnesses, and imperfect perceptions of the real world. Uncertainties can also be inherented from the imprecision of the representation language and media in which the

information is conveyed. It can also be caused by the incompleteness of the information source, and the aggregation or summarization of information from multiple sources [BSAW91].

The toss of a coin may serve as an example. Uncertainties exhibit themselves in the sense that although it is known that either heads or tails are the outcomes, the actual outcomes are open. Other examples closely related to our daily lives including physician's diagnoses in some cases and weatherman's (weatherperson in today's phraseology) forecasts.

It has been pointed out in [Grah91] that if a database management system is truly to fulfill its purpose it must then know how to treat incomplete information. The research activities in this area have focused mainly on the extension of the relational model proposed by Codd [Codd70]. This is due to the mathematical foundation and uniformity of the relational approach and the fact that the value based database systems (relational) will remain the dominate approach in most applications. The main vehicle for this purpose has been the so-called *null value* in its various manifestations (e.g., [AbKG87], [Bisk81], [Bisk83], [Codd75], [Codd79], [GaNP92], [Gran79], [Gran91], [Imie89], [ImLi84], [KuKu91], [Lips79], [LiSu88], [LiS90a], [LiS90b], [LiSu91], [LiZh91], [LiZh92], [MiGr88], [ReFS92], [Reit78], [Reit84], [Vass79], [Zani84], [ZhLi92], [ZhLi93]).

With the invention of extended relational models with incomplete information comes the inevitable necessity for solving the following three major problems: (1) how to correctly process queries issued against the extended relations with incomplete information, (2) how to correctly update the extended relations with incomplete information, and (3) how to correctly handle data dependencies within the context of incomplete information under the extended relations? A little reflection will reveal, however, that all these problems convergence to a common underlying question: how to meaningfully interpret and process incomplete information? The remaining chapters of this dissertation will address and attempt to answer the first two questions.

The next chapter provides some background information and examines previous work in this

area which is required to describe the research work presented in this dissertation.

Chapter 3 extends the work on extended relational model with indefinite and maybe information introduced by Liu and Sunderraman in [LiS90b]. In this model, a data structure called *I-table* was defined for capturing definite, indefinite, and maybe information. According to [LiS90b], an I-table consists of three components, one for each of the three types of information it represents. The *definite component* consists of a set of tuples, each of which is known to be true. The *indefinite component* consists of a set of tuple sets, each tuple set is known to be true. However, it is not known which tuple(s) is(are) true within each tuple set. The *maybe component* also consists of a set of tuples, each of which may be true.

In [LiS90b], five primitive relational operators (viz., selection, projection, cartisian product, union, and difference) were defined on the I-tables of the extended relational database model with indefinite and maybe information. However, another relational operator, natural join, was not studied. Natural join is an important and frequently used operator, due to the necessity of decomposing relations for normalization, and the decomposed relations sometimes need to be natural-joined together. Practically speaking, it is important that natural join operations are carried out efficiently. Considerable effort has gone into optimizing join operations with regard to the conventional relational database model (e.g., [CaRS89], [ChFM87], [ChMG87], [Hill86], [HoWo89], [IoKa90], [JaKo84], [Jark87], [Kim80a], [Kim80b], [MaZd87], [RaPr87], [Seli86], [Swam89], [ThRN87], and [Yaos79]). In Chapter 3, the natural join operation on I-tables of the extended relational model with indefinite and maybe information is defined in a semantically correct manner and an algorithm for computing natural joins using only, in general, a linear number of pair-up operations and, in the worst case, a polynomial number of pair-up operations and a linear number of block accesses with respect to the size of I-tables is presented.

Imielinski and Lipski in [ImLi84] established a framework on the semantic meaningfulness of extended relational models and formalized precise conditions to ensure soundness and

completeness of extended relational operators. Extended relational systems that satisfy these conditions were called representation systems. Three extended relational models (e.g., *Codd tables* [Bisk81], [Codd79]), *naive tables* [ImLi84], and *conditional tables* [ImLi84] introduced for dealing with incomplete information in the form of null values were studied based on these conditions. The results indicated that, with the exception of conditional tables, none of the systems was a representation system with respective to the primitive relational operators (viz., selection, projection, cartisian product, union, and difference) and the join operation. Abiteboul and Grahne broadened this study to include update operations and concluded that only the conditional table possessed the ability to manage updates in a semantically correct manner [AbGr85].

The issue of updating incomplete databases, which has been neglected to a certain extent, is as important and delicate as that of querying them. As a matter of fact, it has been pointed out in [AbGr85] that these two aspects are intimately related as the ability to answer queries assumes the capability to enter incomplete information into databases, and hopefully to remove uncertainties as the result of updates [Win86a]. Therefore, it is fundamental to understand the impact of updates on incomplete information. [AbGr85], [Chol88], [FaUV83], [MaWa87], [Reit88], [Win86a], [Win86b], [Win86c], and [Wins90] are among the works which investigate the subject of updates.

It has been proven that I-tables are able to correctly manage all the primitive relational operators and the join operator in a semantically correct manner ([LiS90b], [LiZh91]). In Chapter 4 we will further our exploration by extending update operations to I-tables and then examining the ability of I-tables for managing update operations in a semantically correct way.

Chapter 5 furthers the study presented in Chapter 4 by extending update operations to a generalization of I-tables called M-tables in a semantically correct manner. M-tables are capable of representing more general forms of disjunctive information such as $P_1(t_1) \vee \cdots \vee P_n(t_n)$,

where the P_i's could be different relation names representing different relations or predicates. Two dimensions of indefinite information can be represented in an M-table: vertical and horizontal. Vertical indefinite information refers to disjunctive information within a single relation and are represented via M-tuple sets of arity two or more of the sure component in M-tables. On the other hand, horizontal indefinite information alludes to uncertainties among different relations and are represented via M-tuple sets with at least two tuples over different relational schemes.

One of the problems associated with various null value approaches proposed in the literature is that they only allow a limited form of incomplete information to be represented. A more serious/fatal deficiency of some of the proposed models for null values lies in the fact that they do not support the relational operators of selection, projection, cartisian product, union, difference, and join in a semantically correct manner.

One direction for further research therefore consists of examining the possibility of generalizing various null value approaches to represent the most general forms of incomplete information. The limitation of the expressive power of the existing null value approaches is resulted from their underlying treatment of incomplete information on the attribute level. Therefore, a more general model should handle incomplete information on the tuple level.

The focus of the research to be discussed in Chapter 6 is that of extending the relational model to represent exclusively disjunctive information. That is, disjunctions of the form $P_1 \mid P_2 \mid \cdots \mid P_n$, where $\mid$ denotes a generalized logical exclusive *or* indicating that exactly one of P_i's can be true. The generalization is necessary since the regular logical exclusive or $\bar{\vee}$ is defined based on the number of true/false values. For example, $1 \bar{\vee} 1 \bar{\vee} 0 = 0$ while $1 \bar{\vee} 1 \bar{\vee} 1 = 1$. Note also that the generalized exclusive *or* represents an *if* $\cdots$ *then* $\cdots$ conjunction in the sense that if P_i is true and $P_1 \mid P_2 \mid \cdots \mid P_i \mid \cdots \mid P_n$ is true, then all the P_j's other than P_i are false, $1 \le i \le n$, $1 \le j \le n$ and $j \ne i$. Specifically, an extended relational

model called *E-table* for capturing exclusively disjunctive information is introduced. Informally, an *E-table* structure T over a relational scheme $R=<A_1, A_2, \cdots, A_k>$ consists of two components, the definite component T_D and the (exclusively) indefinite component T_E. The definite component is a set of (conjunctive) tuples over R, each of which is known to be true. The indefinite component consists of (conjunctive) sets of tuple sets such that each set of tuple sets is known to be true. However, it is not known which tuple set is true within each set of tuple sets. Note that the exclusive indefiniteness implies that one and only one tuple set within a set of tuple sets can be true.

With the introduction of E-tables, the formalization of the semantics of relational operators then becomes the most important issue. A major portion of Chapter 6 is dedicated to this issue. Specifically, the relational operators of selection, projection, cartisian product, difference, union, and intersection are extended to E-tables in a semantically correct manner. All of the extended operators are faithful in the sense that they reduce to the usual relational operators when the E-tables consist of only complete information.

In summary, the issues need to be examined and resolved towards a relational representation of exclusively disjunctive information are as follows:

1. devising an extension of the relational model so that exclusively disjunctive information can be represented,
2. to pave the way for the subject to be studied in (3), the following two problems need to be resolved first:
    - identifying conditions under which redundancies arise due to the presence of exclusively disjunctive information and proposing an operator for removing redundancies,
    - examing the implication of the Closed World Assumption under the context of

exclusively disjunctive information. That is, investigating the relationships between negation and exclusive indefiniteness,

3. extending relational operators to the extended model in a semantically correct manner. Note that this objective is closely tied with the first issue since the extended model should facilitate this requirement.

Finally, Chapter 7 concludes this dissertation with discussions on the directions for further research in the area of incomplete information modeling. In particular, it provides a sketch of a relational model *IE-table* (*I*nclusive and *E*xclusive table) for representing both inclusively and exclusively disjunctive information. It should be clear by now that I-tables model incomplete information in the form of inclusive disjunction, while E-tables model exclusively disjunctive type of incomplete information. Naturally, the logical question then is whether it is possible to represent both inclusively and exclusively disjunctive information under the relational approach. The research in this area presented in the literature [MiGr88] employs the set concept to represent incomplete information. [MiGr88]'s approach allows generalized sets (representing inclusively disjunctive information), disjunctive sets (representing exclusively disjunctive information), and collective sets (representing set values - that is every value inside a collective set is an actual value) as attribute values. The distinction of the corresponding type of each set can be tagged by a mapping from sets to values indicating the associated set types or can be stored in an extraneous tagging field as a part of a relation itself. The IE-table model offers a different viewpoint and adopts a uniformed relational approach without any extraneous tagging functions or attribute fields. The inspiration for IE-tables comes from the E-table model and the main idea is derived from the fact that, logically, inclusive disjunctions can be represented in terms of conjunctions of exclusive disjunctions. For instance, $a \vee b$ is equivalent to $a \mid b \mid (a \wedge b)$. Therefore, inclusive disjunctions can be explicitly represented as conjunctions of exclusive disjunctions, where each conjunction represents a combination of possibilities or

potential real world truth. Since this is an on going research, only preliminary results are presented.

2. BACKGROUND

This chapter first gives an overview and discusses the future trends of the database technology in general. Then a brief survey of the research in the incomplete database area is offered.

2.1 Future Information Systems - Characteristics and Works To Be Done

It has been noted in the literature (e.g., [Kers90], [MiWi86], [MyBr91], [PaZe91], and [Smit86]) that the next generation of computerized information systems must have not only the ability to store and to access but also to reason about a large volume of information and knowledge. This new breed of information systems centers on an exciting confluence of many technologies and must be developed using an interdisciplinary approach and further developments in the field of data modeling. In addition to being intelligent, it should also exhibit the characteristics of being distributed (e.g., being able to interface/integrate with a heterogeneous information processing environment consisting of various database management systems) and being objected-oriented. Generally speaking, the trend is to consolidate the information system technology with the advancements in other areas so that future information systems will become an inseparable component of the future information age and technology rather than a segregated island.

From the intelligence perspective, the central task of the next generation information systems is to integrate intelligent systems with the database technology. As observed in [Kers90], [MiWi86], [MyBr91], [PaZe91], and [Smit86], many intelligent systems in the form of expert systems have been successfully developed and employed for many years for solving a variety of complex problems (e.g., medical diagnosis, VLSI design, and CAD/CAM) which require human expertise. However, the majority of current expert system applications are restricted to limited

data sets and have no facilities for data sharing and performing sophisticated data management activities which are demanded by many applications. These capabilities can be achieved by the integration of intelligent systems and database management systems. When database technology is appropriately combined with intelligent systems it offers the potential for allowing knowledge-bases to be shared amongst several applications. It also provides facilities for manipulating persistent data as well as persistent knowledge.

The database and intelligent-based systems are unfortunately at present quite distinct, as noted in [PaZe91]. Conventional database management systems have been developed to store, maintain, and access large amount of data representing facts organized in well formulated structures and to support transaction intensive data processing applications. In contrast, intelligent-based technology has focused on an increased expressiveness, or the ability to represent many different and complex types of data and their relationships with increasing depth and granularity. Furthermore, Database systems are evolving in the direction of capturing and expressing more semantics in their conceptual schemas, while intelligent systems are trying to deal with applications that require an increasing amount of facts to be stored and managed. The integration of these two technologies should lead to systems capable of managing a large body of complex knowledge in an integrated way ([Kers90], [MiWi86], [MyBr91], [PaZe91], and [Smit86]).

Today's database management systems are often stand-alone. However, the geographically diversified nature of the future information age requires that the next generation information systems be able to process information residing in widely apart physical locations. Moreover, the continual decline of hardware cost, the proliferation of computers, the rapid advancement in communication and networking technology reveals that the future computing environment is to be composed of a large number of workstations and mainframes. This phenomenon has propelled the emergence of distributed information technology. The major impetus of

distributed processing is in the direction of distributed database systems and information technology. This work calls for the integrating of existing heterogeneous applications and resources and concerns with logically integrating this pool of heterogeneous information systems into what appears to be a single logical facility. The benefits envisioned by the development of such a distributed environment are the abilities to collectively manage large volumes of corporate-wide information to lower production and maintenance costs so as to obtain the best possible leverage from pre-existing information intensive applications and resources and to form a synergy of information systems working together in a cooperative manner ([BrCe91], [GuMa87], [HMMS92], [LaLy89], [PaZe91], [PeRR91]).

As far as object-oriented methodology is concerned, it has already penetrated into many subdisciplines of computer science and the database field is no exception. Object-Oriented database systems are receiving increasing attention from both experimental and theoretical standpoints. The object-oriented approach for database systems appears to be very promising and is claiming advantages over the more traditional approaches. The main justifications and reasons are being the modularity and reusability of the object-oriented method and the suitability of object models for treating complex and irregular objects (e.g., in engineering systems), their conceptual naturalness (e.g., for multi-media office systems), and their consonance with strong trends in programming languages and software engineering. The object-oriented database system work is being heavily influenced by the object-oriented paradigm from programming languages. The concept of object-orientation provides the ability to model the real world not only in terms of its structural aspect (e.g., object class definitions) but also in terms of its behavioral or dynamic aspect (e.g., the encapsulation of the methods or operations associated with objects). The main issue in this area is to precisely define what an object-oriented database system is and identify its desired characteristics. The integration of object-orientation and database should greatly enhance the usefulness of the database approach and at the same time

combining the strength of traditional database systems with the salient features of object-oriented approach ([Banc92], [Beec92], [Comm90], [CaRo86], [Ditt86], [GuMa87], [Heue88], [HMMS92], [Kimw90], [Maie86]).

Generally speaking, database systems themselves are evolving in the direction of incorporating and expressing more semantics and a wider range of real world information in the conceptual level. This is evidented by the many data models proposed over the last two decades. In the seventies the relational model received much attention. In the eighties semantic models and non-first-normal form models were the focus. Towards the end of the 80's, object-oriented models were proposed and is continuing to be advocated. An important strand in the data modeling aspect is the modeling of indefinite/uncertain information. The research in this area started in the late seventies and is still receiving a lot of attention. Many problems remain open and wait to be resolved.

In essence, the next generation information systems need further research. The focus of the research work explored in this dissertation is related to the data modeling aspect associated with this next generation of information systems. Specifically, the recurrent and prominent problem of modeling null values in the form of inclusively and exclusively disjunctive forms within the realm of the relational model is addressed. Therefore, a brief survey of the research in the incomplete database modeling arena is presented in the next section. The survey is not intended to be exhaustive. It rather summarizes some representative works in this area.

2.2 In complete Information and Relational Models

The relational database model proposed by Codd in [Codd70] has so far attracted, due to its mathematical foundation and uniformity, a great deal of research activities. Among such developments is the attempt to incorporate a wider range of real world information into the relational model. One aspect of such real world information which has been studied extensively

is that of imprecise data (i.e., incomplete perception of the real world). Two closely related yet divergent types of imprecise information have captivated attentions. They are known as vague information and incomplete information.

The technique for incorporating vagueness in the relational model, based on the fuzzy set theory and fuzzy logic proposed by Zedah [Zeda65], was introduced by Buckles and Petry in their classic works ([BuP82a], [BuP82b], and [BuP82c]). Their fuzzy relational model constitutes of tuple components which are members of power sets over their corresponding domains, a similarity relation for each domain, and possibly fuzzy numbers as domain values. In the fuzzy relational model, a singleton set expresses an absolute truth. On the other hand, the increase of vagueness is signified by a larger number of indistinguishable elements in the set while the decrease of vagueness is indicated by a smaller number of elements in the set. Their work was further investigated in [ShMe90].

On the side of the incomplete database camp, the main vehicle for representing incomplete information has been the so-called *null* values in its various manifestations (e.g., range values, partial values, incomplete values, disjunctive sets, collective sets, etc.). The notion of null values have evolved from completely unknown null values (i.e., any of the values in the domain of the attribute) to partially unknown range values (i.e., any of the values within a specific subset of the attribute domain) ([Bisk81], [Bisk83], [Codd75], [Codd79], [ImLi84], [Lips79], [Reit84], [Vass79], and [Zani84]). Partial values was studied in [Gran79] where by allowing nonatomic values, partial numerical values are represented by a pair of numbers denoting the lower and the upper bound of the range and partial character strings are embedded with the special character "-" (e.g., incompl-te). Another aspects of the null values is that of disjunctive and maybe information. Inclusively disjunctive information (e.g., P is true, or Q is true, or both P and Q are true) were explored in [GaMN84], [Lips79], [LiSu88], [LiS90a], [LiS90b], [LiSu91], [LiZh91], [LiZh92], [MiGr88], and [Reit84]. Exclusively disjunctive information (e.g., either P is true, or

Q is true) was discussed in [MiGr88] and [ZhLi93]. Maybe information was investigated in [Bisk83], [Lips79], [LiSu88], [LiS90a], [LiS90b], [LiSu91], and [MiPe84].

The remainder of this section is organized in the following fashion: First, two works that have been recognized to have had profound influence on the development of incomplete databases, namely the model and proof theoretic views of relational databases and the semantic correctness of the extended relational operators on extended relational models with null values are discussed. The presentation of these issues are preluded by the discussion on the strong relationship between first-order logic and relational databases. Then the problem of negative information in relational databases is presented. Finally, various extended relational models representing null values/incomplete information are presented.

2.2.1 The model and proof theoretic views of relational databases

Reiter in [Reit84] explored the close relationship between the relational database theory and the logic paradigm and offered two distinct views of databases and generalized relational theory with null values and disjunctive information in the context of first order logic. In this section, the close relationship between first-order logic and relational databases and the model-theoretic and proof-theoretic views of relational databases are formally recapitulated. The material is extracted from [Reit84].

2.2.1.1 First-order logic

A first-order language F is specified by a pair (A, W), where A is a set of symbols and W a set of well-formed formulas constructed using the symbols of A. A consists of the following:

1. A set of variables such as x, y, z, x_1, y_1, and z_1. There must be infinitely many of these,

2. A set of constants such as *a*, *b*, *c*, *Smith*, and *CS*101. There must be zero or more of these (possible infinitely many),

3. A set of predicates such as *P*, *Q*, *R*, *TEACH*, and *PARTS* (each predicate has an arity *n* denoting the number of arguments it takes). There must be at least one of these (possibly infinitely many),

4. Punctuation signs such as "(", ")", and ",", and

5. Logical operators $\rightarrow$, $\wedge$, $\vee$, $\neg$, and $\equiv$.

In order to define *W*, the concept of terms and atomic formulas are necessary. A variable or a constant of *A* is a term. If *P* is an n-ary predicate of *A* and $t_1, t_2, \cdots, t_n$ are terms, then $P(t_1, t_2, \ldots, t_n)$ is an atomic formula. $P(t_1, t_2, \ldots, t_n)$ is a ground atomic formula if and only if $t_1, t_2, \ldots, t_n$ are all constants. *W* is to be constructed according to the following:

1. An atomic formula is a well-formed formula,

2. If W_1 and W_2 are well-formed formulas, then so are $W_1 \wedge W_2$, $W_1 \vee W_2$, $\neg W_1$, $W_1 \rightarrow W_2$, and $W_1 \equiv W_2$,

3. If *W* is a well-formed formula, then so is (W), and

4. If *x* is a variable and *W* is a well-formed formula, then $(\exists x)(W)$ and $(\forall x)(W)$ are also well-formed formulas.

A relational language is a first-order language (A, W) with *A* having the following properties:

1. There are only finitely many constants in *A*, but at least one,

2. There are only finitely many predicates in *A*,

3. There is a special predicate "=" which functions as equality, and

4. There is a distinguished subset of *A*, possibly empty, of unary predicates. Such unary

predicates are called simple types and are used to model the concept of the domains of relations in databases.

An interpretation I for a first order language $F=(A,W)$ is a triple (D,K,E) where

1. D is a non empty set, called the domain of I, that is, the range of the variables of A,
2. K is a mapping from the constants of A to D (i.e., $K(c)\in D$, for each constant c of A),
3. E is a mapping from the predicates of A into sets of tuples of elements of D (i.e., $E(P)\subseteq D^n$ for each n-ary predicate symbol P). $E(P)$ is called the extension of P in the interpretation I.

An interpretation $I=(D,K,E)$ for a relational language $R=(A,W)$ is a relational interpretation for R if and only if:

1. K is a one-one and onto mapping, and
2. $E(=)=\{(d,d)\,|\,d\in D\}$.

Consider a relational language $R=(A,W)$, where A contains the following:

1. Constants: A, B, C, a, b, c, d, $CS100$, $CS200$, $P100$, and $P200$,
2. Predicates: $TEACHER(.)$, $COURSE(.)$, $STUDENT(.)$, $TEACH(.,.)$, $ENROLLED(.,.)$, and $=(.,.)$, and
3. Simple Types: $TEACHER(.)$, $COURSE(.)$, $STUDENT(.)$.

Then the following interpretation $I=(D,K,E)$ is a relational interpretation for R:

1. $D=\{A, B, C, a, b, c, d, CS100, CS200, P100, P200\}$,
2. K maps the constant symbols into the corresponding domain elements, and

E is shown in Figure 2.1.

TEACHER	COURSE	STUDENT	TEACH	ENROLLED	=
A	CS100	a	A CS100	a CS100	A A
B	CS200	b	A CS200	a CS100	B B
C	P100	c	B P100	b CS100	C C
	P200	d	C P200	c P100	a a
				d CS200	b b
				d P200	c c
					d d
					CS100 CS100
					CS200 CS200
					P100 P100
					P200 P200

Figure 2.1: A Mapping from the Predicates of A into Sets of Tuples of D

Given an interpretation $I=(D,K,E)$ for a first-order language $F=(A,W)$, let ρ be a mapping from the variables of A into D (ρ is called an environment for the variables of A). For a given environment ρ, define a mapping:

$||\ ||_I^{\rho}: terms \rightarrow D$ as follows:

$||c||_I^{\rho} = K(c)$ for each constant symbol $c \in A$, and

$||x||_I^{\rho} = \rho(x)$ for each variable $x \in A$.

The truth value of a well-formed formula in an interpretation I and environment ρ is defined as follows:

1. $P(t_1, \ldots, t_n)$ is true in $<I,\rho>$ if and only if $<||t_1||_I^{\rho}, \ldots, ||t_n||_I^{\rho}> \in E(P)$,

2. $W_1 \wedge W_2$ is true in $<I,\rho>$ if and only if both W_1 and W_2 are true in $<I,\rho>$,

3. $W_1 \vee W_2$ is true $<I,\rho>$ if and only if either W_1 or W_2 is true in $<I,\rho>$,

4. $\neg W_1$ is true $<I,\rho>$ if and only if W_1 is not true in $<I,\rho>$,

5. $W_1 \rightarrow W_2$ is true $<I,\rho>$ if and only if $\neg W_1 \vee W_2$ is true in $<I,\rho>$,

6. $W_1 \equiv W_2$ is true $<I,\rho>$ if and only if both $W_1 \rightarrow W_2$ and $W_2 \rightarrow W_1$ are true in $<I,\rho>$,

7. $(\forall x)(W)$ is true in $<I,\rho>$ if and only if for all $d \in D$, W is true in $<I,\rho'>$, where ρ' is exactly the same as ρ with one exception, ρ' now maps x to d,

8. $(\exists x)(W)$ is true in $<I,\rho>$ if and only if $\neg(\forall x)(\neg W)$ is true in $<I,\rho>$.

An interpretation I is a model of a well-formed formula W if and only if W is true in I. The interpretation in the previous example is a model for each of the following formulas:

1. $(\forall x)(\forall y)[TEACH(x,y) \rightarrow TEACHER(x) \wedge COURSE(y)]$

2. $(\forall x)(\forall y)[ENROLLED(x,y) \rightarrow STUDENT(x) \wedge COURSE(y)]$

3. $(\forall x)[COURSE(x) \rightarrow (\exists y)(TEACHER(y) \wedge TEACH(y,x))]$

4. $(\forall x)[TEACHER(x) \rightarrow (\exists y)(COURSE(y) \wedge TEACH(x,y))]$

A first order query language is defined relative to a given relational language $R=<A,W>$. Specifically, a query for R is any expression of the form $<x_1/\tau_1, \ldots, x_n/\tau_n \mid Y(x_1, \ldots, x_n)>$ where each τ_i is a type composed of simple types of A and $Y(x_1, \ldots, x_n) \in W$. Moreover, the only free variables of $Y(x_1, \ldots, x_n)$ are among $x_1, \ldots, x_n$ and all of the quantifiers occurring in $Y(x_1,\ldots,x_n)$ are type restricted quantifiers. $<x_1/\tau_1, \ldots, x_n/\tau_n \mid Y(x_1, \ldots, x_n)>$ denotes the set of all tuples of constants $c_1,\ldots,c_n$ such that each c_i satisfies the type τ_i and the database satisfies $Y(c_1, \ldots, c_n)$.

The following are examples of queries applicable to the previous example:

1. The query "Who teaches *P100*?" is expressed as:

   $<x/TEACHER \mid TEACH(x,P100)>$.

2. The query "Who are all of *A*'s student?" is expressed as:

   $<x/STUDENT \mid (\exists y/COURSE)(TEACH(A,y) \wedge ENROLLED(x,y)>$.

2.2.1.2 The model-theoretic view of relational databases

According to Reiter, a database can be viewed as a particular kind of first order interpretation, and the query evaluation is a process of truth functional evaluation of first order formulas with respect to this interpretation. At the same time, integrity constraints can be viewed as first order formulas and a database satisfies the integrity constraints if and only if the constraints is true with respect to the database as interpretation. That is, the interpretation must be a model of the integrity constraints. Therefore, a database is a model of some set of integrity constraints and a query is some formula to be evaluated with respect to this model. This view of the database is called the model theoretic view. Historically, this point of view is the prevailing notion taken by database researchers.

Under the model-theoretic view, a relational database is defined as a triple $DB=(R,I,IC)$, where R is the set of symbols, T is a relational interpretation, and IC is a set of well-formed formulas called integrity constraints. The integrity constraints IC are said to be satisfied if and only if for each well-formed formula w of IC, w is true in I.

For each predicate symbol P distinct from =, IC must contain

$$(\forall x_1)...(\forall x_n)(P(x_1,\ldots,x_n)\rightarrow\tau_1(x_1) \wedge \cdots \wedge \tau_n(x_n))$$

where $\tau_1,\ldots,\tau_n$ are simple types and are referred to as the domains of P.

For each predicate symbol P other than =, $E(P)$ corresponds to a relation.

If $Q=<x/\tau \mid W(x)>$ is a query applicable to the database represented as a first-order theory, then an n-tuple $<c_1,\ldots,c_n>$ of constants in R is an answer to Q if and only if

1. $\tau_i(c_i)$ is true, $i=1,2,...,n$, and

2. $W(<c_1,\ldots,c_n>)$ is true in I.

2.2.1.3 The proof-theoretic view of relational databases

Instead of viewing the interpretation I as a set of tables, think of it as a set of ground atomic formulas. This view of database is called the proof-theoretic view which considers a database as a set of first order formulas rather than a model. Under the proof-theoretic view, queries are formulas to be proven, given the database as premises. Satisfaction of integrity constraints are enforced such that they can be proven from the set of first order formulas. For instance, with respect to the education example, think of the interpretation as being specified by the following ground atomic formulas:

$$\{TEACHER(A), TEACHER(B), TEACHER(C), \ldots,$$

$$ENROLLED(d,P100), ENROLLED(d,P200), \cdots\}$$

Viewing this set as a first order theory T (i.e., as a set of well formed formulas of the underlying first order language), there are many formulas that can be proven given T as premises. For example, with respect to the previous example, we have the following:

$T \vdash ENROLLED(c,P100)$

$T \vdash ENROLLED(a,P100) \wedge TEACH(B,P100)$

where $W \vdash w$ means that there is a first order proof of w from premises W.

Let $R=(A,W)$ be a relational language. A first-order theory $T \subseteq W$ is a relational theory of R if and only if it contains the following axioms:

1. Domain Closure Axiom: $(\forall x)(=(x,c_1) \vee \cdots \vee =(x,c_n))$, where $c_1,\ldots,c_n$ are the constants of A,

2. Unique Name Axiom: $\neg(=(c_i,c_j))$, $i,j=1,\ldots,n$, $i\neq j$,

3. Equality Axioms:

   - Reflexivity: $(\forall x)(=(x,x))$,
   - Commutativity: $(\forall x)(\forall y)(=(x,y)\rightarrow=(y,x))$,
   - Transitivity: $(\forall x)(\forall y)(\forall z)(=(x,y)\wedge=(y,z)\rightarrow=(x,z))$,
   - Leibnitz' principle of substitution of equal terms:

     $$(\forall x_1)\ldots(\forall x_n)(\forall y_1)\ldots(\forall u_n)(P(x_1,\ldots,x_n)\wedge=(x_1,y_1)\wedge\cdots\wedge=(x_n,y_n)$$
     $$\rightarrow P(y_1,\ldots,y_n)),$$

4. Completion Axioms: define $C_P=\{<c_1,\ldots,c_n>\mid P(c_1,\ldots,c_n)\in\Delta \text{ and } P\in A\}$, for some set $\Delta\subseteq W$ of ground atomic formulas such that none of whose predicates is the equality predicate. Suppose that $C_P=\{(c_1^1,\ldots,c_n^1),\ldots,(c_1^m,\ldots,c_n^m)\}$, then

   $$(\forall x_1)\ldots(x_n)(P(x_1,\ldots,x_n)\ (=(x_1c_1^1)\wedge\cdots\wedge(x_n,c_n^1))\vee\cdots\vee$$
   $$(=(x_1,c_1^m)\wedge\cdots\wedge=(x_n,c_n^m))), \text{ and}$$

5. The only well-formed formulas of T are of those mentioned in conditions 1 ,2, and 3 above.

Therefore, in the proof-theoretic view, a relational database is defined to be a triple $DB=(R,T,IC)$, where R is a relational language, T is a relational theory, and IC is a set of integrity constraints.

The following theorem [Reit84] shows that the model-theoretic and proof-theoretic views of relational databases are equivalent.

Theorem 2.1.1 Suppose $R=(A,W)$ is a relational language, Then,

1. If T is a relational theory for R, then T has a unique model which is a relational interpretation for R.

2. If I is a relational interpretation for R, then there is a relational theory T such that I is the only model for T.

The proof-theoretic view can be generalized by adding axioms to it. It is easy to incorporate incomplete information, information about events, hierarchies, and inheritance of properties and aggregations into the proof-theoretic view of relational databases [Reit84].

2.2.2 The semantic correctness of extended relational operators

Imielinski and Lipski in [ImLi84] established a framework on the semantic meaningfulness of the extended relational models with null values and formalized precise conditions to ensure soundness and completeness of extended relational operators. A set of relational operators S is sound if no incorrect answers are derivable by using operators in S and is complete if all valid conclusions are obtainable by using operators in S. These conditions are embodied into the definition of the so called representation system in [ImLi84]. Informally, Imielinski and Lipski stipulate that an extended relational operator f on an extended relational table T of an extended relational model with null values is correct if f on T captures the effect of the corresponding regular operator on the various conventional definite relations represented by T.

Formally, suppose that Γ is the set of extended relations capable of representing incomplete information and Σ is the set of regular relations. Define a mapping *REP* which maps an extended relation, T, to elements of Σ. $REP(T)$ is called the information content of T and is essentially a database containing a set of complete instances which are the possible real world truth. Then the notion of correctness for an extended relational operator f can be illustrated by Figure 2.2.

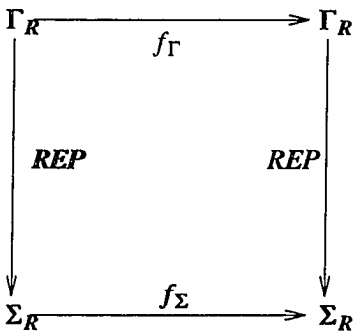


Figure 2.2: Notion of Correctness - Representation System

2.2.3 The closed world assumption

For the discussion that follows a database is viewed as a conjunction of ground atomic formulas (i.e., proof theoretic view). A database is Horn, or definite, if every ground atomic formula is Horn while a ground atomic formula is Horn if it has at most one positive atom when written in its corresponding disjunctive form. On the contrary, a database is non-Horn, or indefinite, if it contains non-Horn ground atomic formulas, which have more than one positive atoms when written in it corresponding disjunctive form.

The Closed World Assumption (CWA) concept was introduced by Reiter for Horn databases in [Reit78]. Under CWA, positive facts, or atomic formulas, are explicitly stored in the database and negative facts are implicitly present provided that their corresponding positive counterpart cannot be proven from the positive facts explicitly stored in the database. Semantically, a negative ground formula is true under CWA if its counterpart is not in the minimal model of the database (a minimal model for Horn database is the intersection of all its models). In other words, negative information can be assumed to be true straightforwardly if its positive counterpart cannot be proven from the database. It has been shown that Horn databases are consistent with CWA. The objective of CWA is to avoid storing potentially overwhelming

amount of negative information explicitly in the database. The opposite of the Closed World Assumption is the Open World Assumption (OWA). Figure 2.3 exemplifies the proceeding discussion based on the following assumptions:

1. $DOMAIN(TEACHER) = \{Smith, Brown, Young\}$,
2. $DOMAIN(COURSE) = \{CS101, CS102\}$, and
3. $DOMAIN(SEMESTER) = \{Fall, Spring, Summer\}$.

Note that the Closed World Assumption makes an assumption about our knowledge about the domain, namely, that we know everything about each predicate of the domain. There are no gaps in out knowledge. For example, if we were ignorant as to whether or not Smith teaches CS102, we could not permit the implication of the opposite by the Closed World Assumption. The implicit representation of negative facts presumes total knowledge about the domain being represented.

However, CWA introduces inconsistencies under non-Horn databases. For instance, let $DB = \{P_a \vee P_b\}$, then both $\overline{P}_a$ and $\overline{P}_b$ are true since they cannot be derived form DB. Consequently, $DB \cup \{\overline{P}_a, \overline{P}_b\}$ is inconsistent. Minker in [Mink82] proposed an extension of the CWA, the Generalized Closed World Assumption (GCWA) for non-Horn databases based on the concept of minimal models. A minimal model for a non-Horn database is a model of that database such that no proper subset of that model is also a model. Semantically, GCWA defines that a negative fact can be assumed to be true if its positive counterpart is not in any minimal models of the underlying first-order theory of its corresponding database. GCWA is consistent with non-Horn databases and is faithful in the sense that it reduces to CWA for Horn databases.

TEACH

TEACHER	COURSE	SEMESTER
Smith	CS101	Fall
Brown	CS102	Fall
Young	CS101	Spring

¬ TEACH

TEACHER	COURSE	SEMESTER
Smith	CS101	Spring
Smith	CS101	Summer
Smith	CS102	Fall
Smith	CS102	Spring
Smith	CS102	Summer
Brown	CS101	Spring
Brown	CS101	Summer
Brown	CS102	Fall
Brown	CS102	Spring
Brown	CS102	Summer
Young	CS101	Fall
Young	CS101	Summer
Young	CS102	Fall
Young	CS102	Spring
Young	CS102	Summer

Figure 2.3: Closed World Assumption

To digress for a moment, very closely related to the notion of the Closed World Assumption is that of negation as failure in logic programming [Clar78]. Logic programming has had significant impact on the development of databases. In particular, it has contributed to the understanding of the semantics of databases and has extended the concept of relational databases [GrMi92]. A survey of negations in logic programming can be found in [Shep87].

2.2.4 Extended relational models with null values

There has been various proposals of extended relational models representing null values. The flavor and spirit of these models can be classified into three basic underlying structure: (1) null values in its original form (i.e, any of the values in the domain), (2) marked null values, and (3) null values in the form of range values/disjunctive sets.

The following is an example of the extended relational model with null values which ranges over the whole domain of the attribute it represents:

TEACH

TEACHER	COURSE	SEMESTER
Smith	@	Fall
Brown	CS101	@
Young	@	@

Figure 2.4: The TEACH Relation with Null Values

The above table represents the following information: (1) Smith teaches a course (however, it is not known which course) in the Fall Semester, (2) Brown teaches the course CS101. However the semester in which Brown teaches CS101 is unknown, and (3) Young teaches a course in one of the semesters yet both the course and semester are unknown. Suppose that the domain of the attribute COURSE contains "CS101" and "CS102" and the possible values for the SEMESTER attribute are the usual Fall, Spring, or Summer. Then this table represents the possible real world truth is shown in Figure 2.5.

Smith CS101 Fall Brown CS101 Fall Young CS101 Fall	Smith CS101 Fall Brown CS101 Fall Young CS101 Spring	Smith CS101 Fall Brown CS101 Fall Young CS101 Summer
Smith CS101 Fall Brown CS101 Fall Young CS102 Fall	Smith CS101 Fall Brown CS101 Fall Young CS102 Spring	Smith CS101 Fall Brown CS101 Fall Young CS102 Summer
Smith CS101 Fall Brown CS101 Spring Young CS101 Fall	Smith CS101 Fall Brown CS101 Spring Young CS101 Spring	Smith CS101 Fall Brown CS101 Spring Young CS101 Summer
Smith CS101 Fall Brown CS101 Spring Young CS102 Fall	Smith CS101 Fall Brown CS101 Spring Young CS102 Spring	Smith CS101 Fall Brown CS101 Spring Young CS102 Summer
Smith CS101 Fall Brown CS101 Summer Young CS101 Fall	Smith CS101 Fall Brown CS101 Summer Young CS101 Spring	Smith CS101 Fall Brown CS101 Summer Young CS101 Summer
Smith CS101 Fall Brown CS101 Summer Young CS102 Fall	Smith CS101 Fall Brown CS101 Summer Young CS102 Spring	Smith CS101 Fall Brown CS101 Summer Young CS102 Summer
Smith CS102 Fall Brown CS101 Fall Young CS101 Fall	Smith CS102 Fall Brown CS101 Fall Young CS101 Spring	Smith CS102 Fall Brown CS101 Fall Young CS101 Summer
Smith CS102 Fall Brown CS101 Fall Young CS102 Fall	Smith CS102 Fall Brown CS101 Fall Young CS102 Spring	Smith CS102 Fall Brown CS101 Fall Young CS102 Summer
Smith CS102 Fall Brown CS101 Spring Young CS101 Fall	Smith CS102 Fall Brown CS101 Spring Young CS101 Spring	Smith CS102 Fall Brown CS101 Spring Young CS101 Summer
Smith CS102 Fall Brown CS101 Spring Young CS102 Fall	Smith CS102 Fall Brown CS101 Spring Young CS102 Spring	Smith CS102 Fall Brown CS101 Spring Young CS102 Summer
Smith CS102 Fall Brown CS101 Summer Young CS101 Fall	Smith CS102 Fall Brown CS101 Summer Young CS101 Spring	Smith CS102 Fall Brown CS101 Summer Young CS101 Summer
Smith CS102 Fall Brown CS101 Summer Young CS102 Fall	Smith CS102 Fall Brown CS101 Summer Young CS102 Spring	Smith CS102 Fall Brown CS101 Summer Young CS102 Summer

Figure 2.5: Possible Real World Truth for the TEACH Relation of Figure 2.4

This set of regular relations is called the information content of the null table as shown in Figure 2.4. In [Maie83], the concept of possibility set is used to represent the information content of a null table and the information content as shown in figure 2.5 is equivalent to the possibility set of $POSS_c(T)$ of [Maie83].

Under this approach, there is only one null value. Therefore, it is impossible to represent, for example, the information that Smith and Young teach same/different courses. This deficiency is resolved by introducing marked null values which are assumed distinct unless they have the same subscript. Figure 2.6 gives an example of a table with marked null values.

This table represents the information, in addition to those deciphered by the extended relation shown in figure 2.4, the fact that Smith and Young teach different courses and Brown and Young teach in the same semester. The information content of this table is shown in Figure 2.7.

TEACH

TEACHER	COURSE	SEMESTER
Smith	$@_1$	Fall
Brown	CS101	$@_2$
Young	$@_3$	$@_2$

Figure 2.6: A Sample Relation with Marked Null Values

Smith CS101 Fall Brown CS101 Fall Young CS101 Fall	Smith CS101 Fall Brown CS101 Spring Young CS101 Spring	Smith CS101 Fall Brown CS101 Summer Young CS101 Summer
Smith CS101 Fall Brown CS101 Fall Young CS102 Fall	Smith CS101 Fall Brown CS101 Spring Young CS102 Spring	Smith CS101 Fall Brown CS101 Summer Young CS102 Summer
Smith CS102 Fall Brown CS101 Fall Young CS101 Fall	Smith CS102 Fall Brown CS101 Spring Young CS101 Spring	Smith CS102 Fall Brown CS101 Summer Young CS101 Summer
Smith CS102 Fall Brown CS101 Fall Young CS102 Fall	Smith CS102 Fall Brown CS101 Spring Young CS102 Spring	Smith CS102 Fall Brown CS101 Summer Young CS102 Summer

Figure 2.7: Possible Real World Truth for the TEACH Relation of Figure 2.6

The null values represented in the above described approaches denote completely unknowns. That is, a null value can be any of the values in the domain of the attribute it is in. Later on, these approaches evolved into a more precise form under which missing values are partially unknown in the sense that they can be any of the values in a specific subset of the attribute domain. This is accomplished by allowing (disjunctive) sets/range values as attribute values. For instance, the table shown in Figure 2.8 represents the possible real world truth as shown by its information content in Figure 2.9.

TEACH

TEACHER	COURSE	SEMESTER
Smith	{CS101, CS102}	Fall
Brown	CS101	{Fall, Spring}
Young	{CS101, CS102}	{Fall, Spring}

Figure 2.8: The TEACH Relation with Range Values

Smith CS101 Fall Brown CS101 Fall Young CS101 Fall	Smith CS101 Fall Brown CS101 Fall Young CS101 Spring	Smith CS101 Fall Brown CS101 Spring Young CS101 Fall	Smith CS101 Fall Brown CS101 Spring Young CS101 Spring
Smith CS101 Fall Brown CS101 Fall Young CS102 Fall	Smith CS101 Fall Brown CS101 Fall Young CS102 Spring	Smith CS101 Fall Brown CS101 Spring Young CS102 Fall	Smith CS101 Fall Brown CS101 Spring Young CS102 Spring
Smith CS102 Fall Brown CS101 Fall Young CS101 Fall	Smith CS102 Fall Brown CS101 Fall Young CS101 Spring	Smith CS102 Fall Brown CS101 Spring Young CS101 Fall	Smith CS102 Fall Brown CS101 Spring Young CS101 Spring
Smith CS102 Fall Brown CS101 Fall Young CS102 Fall	Smith CS102 Fall Brown CS101 Fall Young CS102 Spring	Smith CS102 Fall Brown CS101 Spring Young CS102 Fall	Smith CS102 Fall Brown CS101 Spring Young CS102 Spring

Figure 2.9: Possible Real World Truth for the TEACH Relation of Figure 2.8

2.2.5 Extended relational models with disjunctive information

Recently, an extended relational model, namely *I-table*, for capturing incomplete knowledge appeared as indefinite and maybe information was proposed ([LiSu88], [LiS90a]). Informally, an *I-table* consists of three components, one for each of the three types of information it represents. The *definite component* consists of a set of tuples, each of which is known to be true. The *indefinite component* consists of a set of tuple sets, each tuple set is known to be true. However, it is not known which tuple(s) is(are) true within each tuple set. The *maybe component* also consists of a set of tuples, each of which may be true.

Consider, for example, the familiar database relation **TEACH**. The information that Smith teaches CS101 in Fall can be represented by the definite component as {(Smith, CS101, Fall)}. The information that Brown teaches CS102 either in Fall or Spring, and Young teaches CS101 in either Fall or Spring or Young teaches CS102 in Fall can be represented by the indefinite component as {{(Brown, CS102, Fall),(Brown, CS102, Spring)}, {(Young, CS101, Fall),(Young, CS101, Spring),(Young, CS102, Fall)}}. The information that White may teach CS101 in Spring can be represented by the maybe component as {(White, CS101, Spring)}. The

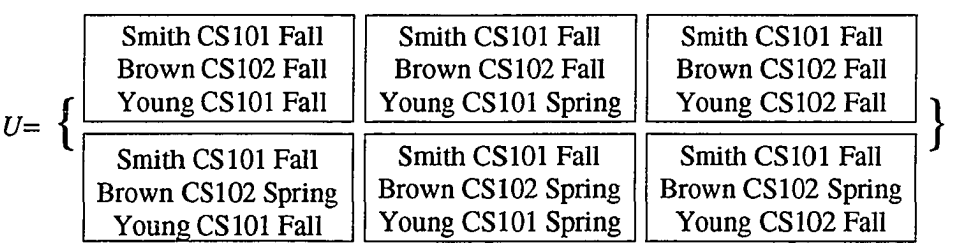
corresponding I-table is shown in Figure 2.10.

The information content associated with an I-table consists of two components: the sure component and the maybe component. The sure component deciphers the information represented by the definite and the indefinite component of an I-table. It represents various definite relations of which at least one is the real world truth. The maybe component accounts for all the maybe tuples in the I-table. The information content of the I-table shown in Figure 2.10 is illustrated in Figure 2.11.

TEACH

TEACHER	COURSE	SEMESTER
Smith	CS101	Fall
Brown	CS102	Fall
Brown	CS102	Spring
Young	CS101	Fall
Young	CS101	Spring
Young	CS102	Fall
White	CS101	Spring

Figure 2.10: A Sample I-table



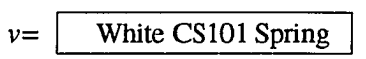


Figure 2.11: The Information Content of the I-table of Figure 2.10

2.2.6 Generalized relational model with disjunctive information

In [LiS90b], a generalized extended relational model, namely the M-table model, was proposed to represent more general forms of incomplete information. As described in the previous section, the I-table model is capable of representing indefinite and maybe information of the form $P(t_1) \vee \cdots \vee P(t_n)$, where all the disjuncts involve the same predicate symbol. M-tables are generalizations of I-tables and are able to represent disjunctive information of the form $P_1(t_1) \vee \cdots \vee P_n(t_n)$, where P_i's, $i = 1,\ldots,n$, could be different predicates.

Two dimensions of indefinite information can be represented in an M-table: vertical and horizontal. Vertical indefinite information refers to disjunctive information within a single relation and are represented via M-tuple sets of arity two or more of the sure component in M-tables. On the other hand, horizontal indefinite information alludes to uncertainties among different relations and are represented via M-tuple sets with at least two tuples over different relational schemes.

Informally, an *M-table* T over a mixed relational, or M-relational scheme $MR = <R_1, R_2, \cdots, R_k>$ is a group of k mixed relations, or M-relations $< r_1, \ldots, r_k >$, where r_i is a relation over $R_i, 1 \le i \le k$, respectively. It consists of two components: the *sure component* T_{sure} and the *maybe component* T_{maybe}. The *sure component* of an M-table is a set of mixed tuple, or M-tuple sets of arities greater than or equal to 1. A mixed tuple, or M-tuple is of the form $<t_1, t_2, \cdots, t_k>$, where t_i is a tuple over relational scheme $R_i, 1 \le i \le k$, and each M-tuple set is known to be true. The *maybe component* consists of a set of mixed tuples, each of which may be true.

By way of example, Figure 2.12 illustrates an M-table *PARENT-CHILDREN* over M-relational scheme *<FATHER-SON, MOTHER-SON, FATHER-DAUGHTER, MOTHER-DAUGHTER>*.

$PARENT-CHILDREN_{sure}$ represents the following information, or ground formulas:

GF1. FATHER-SON(John, Joe)

FATHER	SON	MOTHER	SON	FATHER	DAUGHTER	MOTHER	DAUGHTER
John	Joe						
Chris	Jack	Chris	Jack				
		Mary Nancy	Sam Sam			Mary Nancy	Sam Sam
				Steve Jack	Linda Linda		
Leslie	Pat	Leslie	Pat	Leslie	Pat	Leslie	Pat
Ed	Mike						
Kim	Mark	Kim	Mark				

Figure 2.12: PARENT-CHILDREN M-relation

GF2. FATHER-SON(Chris, Jack) v MOTHER-SON(Chris, Jack)

GF3. MOTHER-SON(Mary, Sam) v MOTHER-DAUGHTER(Mary, Sam) v
MOTHER-SON(Mary, Sam) v MOTHER-DAUGHTER(Mary, Sam)

GF4. FATHER-DAUGHTER(Steve, Linda) v FATHER-DAUGHTER(Jack, Linda)

GF5. FATHER-SON(Leslie, Pat) v MOTHER-SON(Leslie, Pat) v
FATHER-DAUGHTER(Leslie, Pat) v MOTHER-DAUGHTER(Leslie, Pat)

Each of the above five ground formulas are true, while it is not known which ground clause or clauses are true in GF2, GF3, GF4, and GF5. Moreover, the M-tuple sets corresponds to GF2 is an example of horizontal disjunctions across the relations *FATHER_SON* and *MOTHER – SON* while the one associated with GF4 signifies vertical disjunctions within the relation *FATHER – DAUGHTER*. Yet GF3 is derived from the M-tuple set incorporating both vertical and horizontal indefinite information.

PARENT – CHILDREN$_{maybe}$ represents the following ground atomic formulas:

GAF1. FATHER-SON(Ed, Mark)

GAF2. FATHER-SON(Kim, Mark) v MATHER-SON(Kim, Mark)

However, these ground atomic formulas may or may not be true.

The information content associated with an M-table consists of two components: a collection of a set of definite mixed relations, among which at least one is the real world truth, correlating to the minimal models of the underlying first-order theory represented by the sure component of the M-table and a set of mixed maybe tuples. The information content of the M-table shown in Figure 2.12 is illustrated in Figure 2.13.

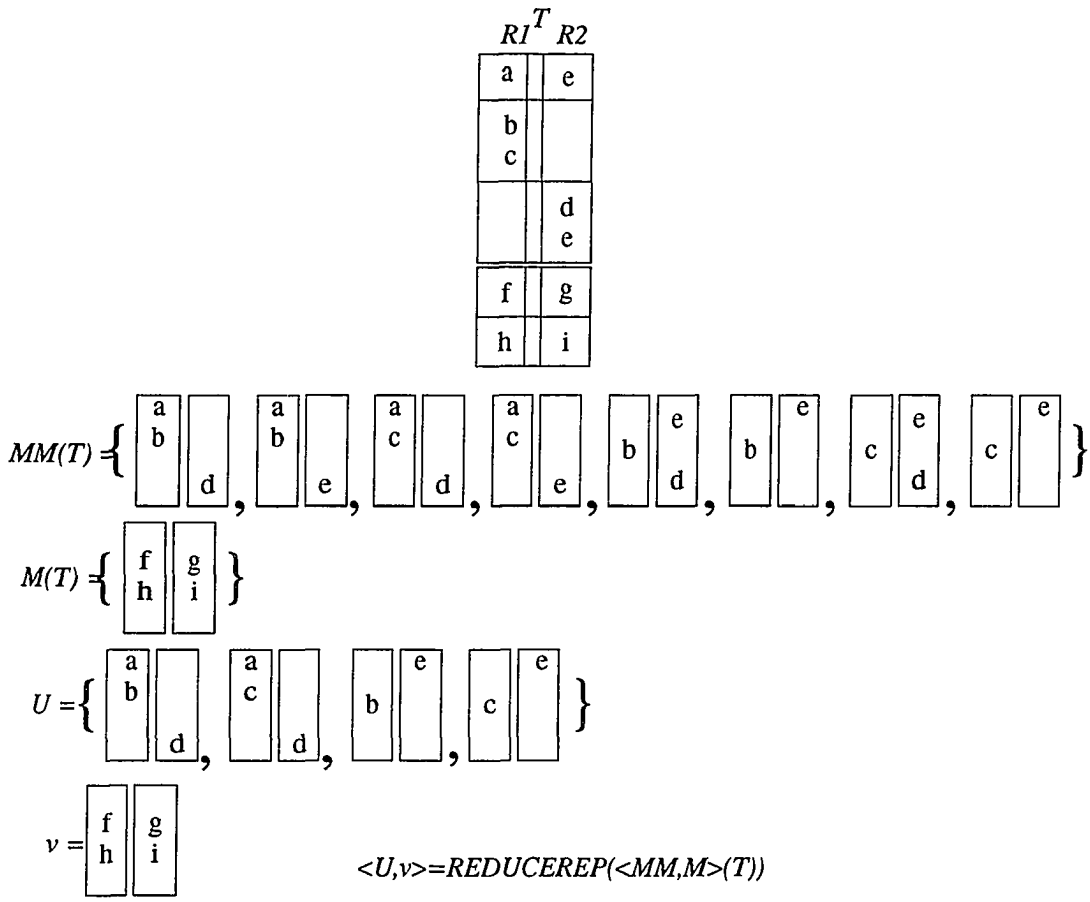


Figure 2.13: The Information Content of the M-table of Figure 2.12

3. NATURAL JOINS AND I-TABLES

In [LiSu90], an extended relational database model with indefinite and maybe information was introduced. Five primitive relational operators (viz., selection, projection, cartisian product, union, and difference) were also defined on I-tables of the extended relational database model with indefinite and maybe information. However, another relational operator, natural join, was not studied in [LiSu90]. Natural join is an important and frequently used operator. This is due to the necessity of decomposing relations for normalization, and the decomposed relations sometimes need to be natural-joined together. Practically speaking, it is important that natural join operations are carried out efficiently. Considerable effort has gone into optimizing join operations with regard to the conventional relational database model (e.g., [CaRS89], [ChFM87], [ChMG87], [Hill86], [HoWo89], [IoKa90], [JaKo84], [Jark87], [Kim80a], [Kim80b], [MaZd87], [RaPr87], [Seli86], [Swam89], [ThRN87], and [Yaos79]).

This chapter extends previous work on I-tables. Specifically, natural joins are defined, in a semantically correct manner, on I-tables. Rudimentary or naive algorithms for computing natural joins on I-tables require an exponential number of pair-up operations and block accesses proportional to the size of I-tables. This is due to the combinatorial nature of natural joins on I-tables. Thus, the problem becomes intractable for large I-tables. In this chapter, an algorithm for computing natural joins under the extended model is proposed. This algorithm reduces the number of pair-up operations to a linear order of complexity in general and in the worst case to a polynomial order of complexity with respect to the size of I-tables. This algorithm also reduces the number of block accesses to a linear order of complexity with respect to the size of I-tables.

This chapter is arranged as follows: in Section 3.1, some background knowledge regarding the extended relational model will be presented. In Section 3.2, the natural join of I-tables is defined, and its correctness is also established. Finally, in Section 3.3, efficient implementation

issues of natural joins on I-tables are discussed.

3.1 Preliminaries

3.1.1 A formal semantics for I-tables

An I-table T over a relational scheme $R=<A_1, \ldots, A_n>$ with domain D_i for each attribute name A_i, $1 \leq i \leq n$, is defined as $T = <T_D, T_I, T_M>$, where

$$T_D \subseteq D_1 \times D_2 \times \cdots \times D_n,$$

$$T_I \subseteq 2^{D_1 \times \cdots \times D_n} - (\{\varnothing\} \cup \{\{t\} \mid t \in D_1 \times \cdots \times D_n\}), \text{ and}$$

$$T_M \subseteq D_1 \times D_2 \times \cdots \times D_n.$$

$T_D, T_I,$ and T_M are referred to as the *definite component*, the *indefinite component*, and the *maybe component* of the I-table T respectively. Each member of T_D and T_M is called a definite tuple and a maybe tuple respectively, and each member of T_I is called an indefinite tuple set. Indefinite tuple sets represent inclusive disjunctions. That is, it is possible for more than one tuple within an indefinite tuple set to be true. Maybe tuples represent information which is either true or false.

Consider, for example, a database relation **ASSIGNMENT** with attributes EMP_NAME and JOB_ASSIGNED. The information that EMP1 and EMP2 are assigned to JOB1 and JOB2 respectively can be represented by the definite component as {(EMP1,JOB1),(EMP2,JOB2)}. The information that either EMP3 or EMP4, or both EMP3 and EMP4, are assigned to JOB3 can be represented by the indefinite component as {{(EMP3,JOB3),(EMP4,JOB3)}}. The information that EMP5 may be assigned to JOB4 can be represented by the maybe component as {(EMP5,JOB4)}. The corresponding I-table is shown in Figure 3.1.

ASSIGNMENT

EMP1	JOB1
EMP2	JOB2
EMP3	JOB3
EMP4	JOB3
EMP5	JOB4

Figure 3.1: An Example of an I-table

There are two sources for the maybe tuples. First, they may be inserted by users. Second, they may be migrated from the *indefinite component* due to updates. For example, if (EMP3, JOB3) were inserted into the I-table as shown in Figure 3.1, then the tuple (EMP4, JOB3) should be moved to the *maybe component.*

The information content of an I-table consists of two components: the sure component and the maybe component. The sure component deciphers the information represented by the definite and the indefinite component of the I-table. It represents various definite relations of which at least one is the real world truth. The maybe component accounts for all the maybe tuples in the I-table. The information content of an I-table is formally defined as follows:

Definition 3.1.1.1: Let R be a relational scheme and T be an I-table over R, then

$\Gamma_R = \{T | T$: *I−table over R* $\}$, and

$\Sigma_R = \{<U,v> | U$: *set of relations over R*, v: *relation over R*$\}$.

The information content of the I-table T is a mapping *REP* from Γ_R to Σ_R such that:

$REP(T) = REDUCEREP(<MM,M>(T))$, where

(1) $<MM,M>$ is a mapping from Γ_R to Σ_R such that:

$<MM,M>(T) = <MM(T),M(T)>$, where

$T=<T_D,T_I,T_M>$ with $T_I=\{w_1,...,w_n\}$, and

$MM(T) = \{T_D \cup \{t_1,\ldots,t_n\} | (\forall i)(1\le i\le n \rightarrow t_i \in w_i)\}$, and $M(T) = T_M$.

(2) *REDUCEREP* is a mapping from Σ_R to Σ_R such that:

$REDUCEREP(<U,v>)=<U^o,v^o>$, where

$U^o = \{r | (r \in U \wedge \neg(\exists r_1)(r_1 \in U \wedge r_1 \subset r)\}$, and

$v^o = \{t | (t \in v \vee (\exists r_1)(\exists r_2)(r_1 \in U \wedge r_2 \in U \wedge r_1 \subset r_2 \wedge t \in r_2 - r_1) \wedge \neg(\exists r)(r \in U^o \wedge t \in r))\}$.

Note that $MM(T)$ consists of all the definite relations represented by the sure components of the I-table and $M(T)$ consists of all the maybe tuples in T_M. On the other hand, U^o is U with all the definite relations which subsume other definite relations removed (therefore, U^o corresponds to the minimal models of the underlying first-order theory), and v^o is v along with some tuples from the definite relations removed from U.

3.1.2 Redundancies in I-tables

Four kinds of redundancies across the components of an I-table have been identified in [LiSu90]. They are:

(1) A definite statement subsumes an indefinite statement. That is, $t \in T_D$ and $w \in T_I$ and $t \in w$.

(2) An indefinite statement subsumes another indefinite statement. That is, $w_1 \in T_I$ and $w_2 \in T_I$ and $w_1 \subset w_2$.

(3) A maybe statement is also a definite statement. That is, $t \in T_M$ and $t \in T_D$.

(4) An indefinite statement subsumes a maybe statement. That is, $t \in T_M$ and $t \in w$ and $w \in T_I$.

The above mentioned redundancies can be removed by the operator called *REDUCE* as defined as follows:

Definition 3.1.2.1 Let T be an I-table. then $REDUCE(T)=T^o$, where

$T_D^o = \{t | t \in T_D\}$,

$$T_I^o = \{w \mid (w \in T_I \wedge \neg(\exists t)(t \in T_D \wedge t \in w) \wedge \neg(\exists w_1)(w_1 \in T_I \wedge w_1 \subset w)\}, \text{ and}$$

$$T_M^o = \{t \mid (t \in A) \wedge (t \notin T_D^o) \wedge \neg(\exists w)(w \in T_I^o \wedge t \in w)\},$$

where A is defined as follows:

$$A = \{t \mid (t \in T_M) \vee (\exists t_1)(\exists w)(t_1 \in T_D \wedge w \in T_I \wedge t_1 \in w \wedge t \in w - \{t_1\}) \vee$$
$$(\exists w_1)(\exists w_2)(w_1 \in T_I \wedge w_2 \in T_I \wedge w_1 \subset w_2 \wedge t \in w_2 - w_1)\}$$

3.1.3 Notions of the correctness of the extended relational algebra

As has been defined earlier, the mapping *REP* maps an I-table, T, over scheme R, to elements of Σ_R. $REP(T)$ consists of two components: U, a set of definite relations of which at least one represents the real world truth, and v, a set of maybe tuples. Consider a relational algebra operator f. In order to extend f to operate on I-tables, it must be assured that the extended operator captures the effect of the corresponding regular operator on the various definite relations represented in the information content of I-tables. This notion of correctness for an extended relational operator f on I-tables is illustrated in Figure 3.2.

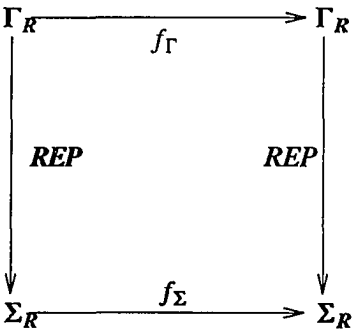


Figure 3.2: The Notion of Correctness

Formally, for each operator f, we first need to define f_Σ on Σ_R and then define f_Γ on Γ_R, that satisfies the following conditions:

$REP(f_\Gamma(T)) = f_\Sigma(REP(T))$, for unary f, and

$REP(f_\Gamma(T_1,T_2)) = f_\Sigma(REP(T_1),REP(T_2))$, for binary f.

3.2 Extended Relational Algebra - Natural Join

In this section, the definitions for extended natural joins on Σ and Γ are first defined. The correctness of the extended natural join is then established. The same symbol ∞ is used to represent the extended natural join for operands on both Σ and Γ. This shall not cause any confusion as the operands should clearly identify whether the natural join is performed on Σ or Γ.

Definition 3.2.1: Let R_1 and R_2 be two relational schemes with some common attributes. Let $<U_1,v_1> \in \Sigma_{R_1}$ and $<U_2,v_2> \in \Sigma_{R_2}$. Then, $<U_1,v_1> \infty <U_2,v_2> = REDUCEREP(<U_1,v_1> \infty^o <U_2,v_2>) = REDUCEREP(<U,v>)$, where

$U = \{r \mid (\exists r_1)(\exists r_2)(r_1 \in POSS(U_1) \wedge r_2 \in POSS(U_2) \wedge r = r_1 \infty r_2)\}$, and

$v = \bigcup_{r \in U_1} (r \infty v_2) \cup \bigcup_{r \in U_2} (r \infty v_1) \cup (v_1 \infty v_2)$.

where $POSS(U) = \{r \mid (\exists k)(1 \le k \le |U| \wedge (\exists r_1) \cdots (\exists r_k)(r_1 \in U \wedge \cdots \wedge r_k \in U \wedge r = r_1 \cup \cdots \cup r_k)\}$.

Notice that $r_1 \subseteq r_2$ implies that $r \infty r_1 \subseteq r \infty r_2$ and $U \subseteq POSS(U)$. By the definition of *REDUCEREP*, the above definition can be simplified into the following equivalent one.

Definition 3.2.2: Let R_1 and R_2 be two relational schemes with some common attributes. Let $<U_1,v_1> \in \Sigma_{R_1}$ and $<U_2,v_2> \in \Sigma_{R_2}$. Then, $<U_1,v_1> \infty <U_2,v_2> = REDUCEREP(<U_1,v_1> \infty^o <U_2,v_2>) = REDUCEREP(<U,v>)$, where

$U = \{r \mid (\exists r_1)(\exists r_2)(r_1 \in U_1 \wedge r_2 \in U_2 \wedge r = r_1 \infty r_2)\}$, and

$$v = \bigcup_{r \in U_1} (r \infty v_2) \cup \bigcup_{r \in U_2} (r \infty v_1) \cup (v_1 \infty v_2).$$

The following theorem shows that natural joins of the elements of Σ commutes with *REDUCEREP*.

Theorem 3.2.1: Let R_1 and R_2 be two relational schemes with some common attributes. Then $<U_1, v_1> \infty <U_2, v_2> = REDUCEREP(<U_1, v_1>) \infty REDUCEREP(<U_2, v_2>)$ for any $<U_1, v_1> \in \Sigma_{R_1}$ and $<U_2, v_2> \in \Sigma_{R_2}$.

Next, we define natural joins on I-tables. Before presenting the definition for natural joins on Γ, a few notations are first defined as follows.

Definition 3.2.3: Let R_1 and R_2 be two relational schemes, r_1 be an I-table over R_1, t be a tuple of r_1, $\{t_1, ..., t_k\}$ be a tuple set of r_1, and C a subset of R_1. Then:

(1) $s = \Pi_C$ (t) denotes the projection of t onto attributes of C. That is, s = t[C].

(2) $\Pi_C\{t_1, ..., t_k\} = \{\Pi_C(t_1), ..., \Pi_C(t_k)\}$.

Now the definition of natural joins on I-tables is given below:

Definition 3.2.4: Let T_1 and T_2 be two I-tables under relational schemes R_1 and R_2 such that $T_I^1 = \{w_1^1, ..., w_m^1\}$ and $T_I^2 = \{w_1^2, ..., w_n^2\}$. Furthermore, let **A** denotes the list of attributes that are common in both R_1 and R_2. That is, $\mathbf{A} = R_1 \cap R_2$, and let **B** denotes the list of attributes that is composed of all the attributes of R_1 and all the attributes of R_2 except those of **A**. That is, $\mathbf{B} = R_1(R_2 - R_1)$. Let

$$E = \{\{t_1, ..., t_m\} \mid (\forall i)(1 \le i \le m \rightarrow t_i \in w_i^1)\}, \text{ and}$$

$$F = \{\{t_1, ..., t_n\} \mid (\forall i)(1 \le i \le n \rightarrow t_i \in w_i^2)\}.$$

Let the elements of E be $E_1, ..., E_e$ and those of F be $F_1, ..., F_f$. Let

$$A_{ij} = \{t \mid (\exists t_1)(\exists t_2)(t_1 \in T_D^1 \wedge t_2 \in F_l \wedge t = \Pi_{\mathbf{B}}(t_1 t_2) \wedge t_1[\mathbf{A}] = t_2[\mathbf{A}]) \vee$$
$$(\exists t_1)(\exists t_2)(t_1 \in E_k \wedge t_2 \in T_D^2 \wedge t = \Pi_{\mathbf{B}}(t_1 t_2) \wedge t_1[\mathbf{A}] = t_2[\mathbf{A}]) \vee$$
$$(\exists t_1)(\exists t_2)(t_1 \in E_k \wedge t_2 \in F_l \wedge t = \Pi_{\mathbf{B}}(t_1 t_2) \wedge t_1[\mathbf{A}] = t_2[\mathbf{A}])\},$$

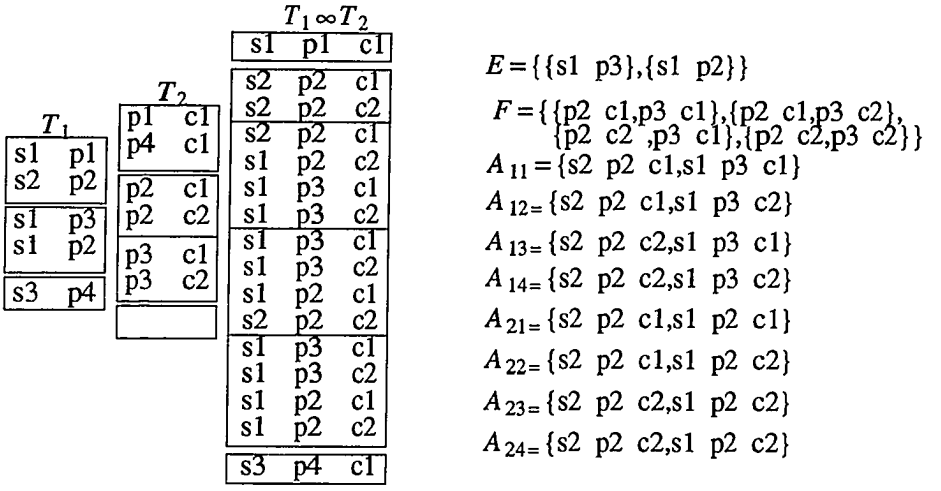


Figure 3.3: Cartisian Product

where $1 \le k \le e$, $1 \le l \le f$, $i = k$ if $e \neq 0$ otherwise $i = 0$, and $j = l$ if $f \neq 0$ otherwise $j = 0$. Let A_1, ..., A_g be the distinct A_{ij}'s. Then $T_1 \infty T_2 = REDUCE(T)$, where T is defined as follows:

$$T_D = \{t \mid (\exists t_1)(\exists t_2)(t_1 \in T_D^1 \wedge t_2 \in T_D^2 \wedge t = \Pi_{\mathbf{B}}(t_1 t_2) \wedge t_1[\mathbf{A}] = t_2[\mathbf{A}])\},$$

$$T_I = \{w \mid (\exists t_1)\ldots(\exists t_g)(t_1 \in A_1 \wedge \cdots \wedge t_g \in A_g \wedge w = \{t_1, \ldots, t_g\})\}, \text{ and}$$

$$\begin{aligned} T_M = \{t \mid & (\exists t_1)(\exists t_2)(t_1 \in T_D^1 \wedge t_2 \in T_M^2 \wedge t = \Pi_{\mathbf{B}}(t_1 t_2) \wedge t_1[\mathbf{A}] = t_2[\mathbf{A}]) \vee \\ & (\exists t_1)(\exists t_2)(t_1 \in T_M^1 \wedge t_2 \in T_D^2 \wedge t = \Pi_{\mathbf{B}}(t_1 t_2) \wedge t_1[\mathbf{A}] = t_2[\mathbf{A}]) \vee \\ & (\exists t_1)(\exists t_2)(t_1 \in T_M^1 \wedge t_2 \in T_M^2 \wedge t = \Pi_{\mathbf{B}}(t_1 t_2) \wedge t_1[\mathbf{A}] = t_2[\mathbf{A}]) \vee \\ & (\exists w)(\exists t_1)(w = \{t_2, \ldots, t_k\} \in T_I^1 \wedge t_1 \in t_M^2 \wedge \\ & \quad ((\exists i) 2 \le i \le k \wedge t = \Pi_{\mathbf{B}}(t_i t_1) \wedge t_i[\mathbf{A}] = t_1[\mathbf{A}])) \vee \\ & (\exists t_1)(\exists w)(t_1 \in T_M^1) \wedge (w = \{t_2, \ldots, t_k\} \in T_I^2 \wedge \\ & \quad ((\exists i) 2 \le i \le k \wedge t = \Pi_{\mathbf{B}}(t_i t_1) \wedge t_i[\mathbf{A}] = t_1[\mathbf{A}]))\}. \end{aligned}$$

Definition 3.2.4 is exemplified in Figure 3.3.

The following theorem establishes the correctness of the extended natural join.

Theorem 3.2.2: Let T_1 and T_2 be two I-tables under relational schemes R_1 and R_2 such that $T_I^1 = \{w_1^1, \ldots, w_m^1\}$ and $T_I^2 = \{w_1^2, \ldots, w_n^2\}$. Then $REP(T_1 \infty T_2) = REP(T_1) \infty REP(T_2)$.

Theorem 3.2.2 is also illustrated in Figure 3.3.

3.3 Algorithms for Natural Joins on I-tables

In this section, two algorithms for computing natural joins on I-tables of the extended relational database model with indefinite and maybe information are presented . The performance of the algorithms is measured by the number of pair-up operations and I/O block accesses. The first algorithm stems directly from the definition and is inefficient and expensive as it requires an exponential number of pair-up operations and block accesses proportional to the size of I-tables. The second one improves the first one by adopting the "compare, concatenate, and then permute" strategy and by accessing temporary relations via pointers, thereby reducing

the number of pair-up operations to a linear order of complexity in general and in the worst case to a polynomial order of complexity with respect to the size of I-tables. Furthermore, it reduces the number of block accesses to a linear order of complexity with respect to the size of underlying I-tables.

The algorithms presented assume that the set of attributes to be joined on both I-tables are indexed. It should be clear that this assumption is practical and conforms with performance requirements in the real application environment.

Algorithm 3.3.1: Rudimentary algorithm for computing the natural join of I-tables T_1 and T_2.

Let $T_D^1 = \{t_1^1,...,t_{N_{D_1}}^1\}, T_I^1 = \{w_1^1,...,w_{N_{I_1}}^1\}$, where $w_i^1 = \{t_{i1}^1,...,t_{ik_i^1}^1\}$, $1 \le i \le N_{I_1}$, and $T_M^1 = \{t_1,...,t_{N_{M_1}}\}$. Let $T_D^2 = \{t_1^2,...,t_{N_{D_2}}^2\}$, $T_I^2 = \{w_1^2,...,w_{N_{I_2}}^2\}$, where $w_i^2 = \{t_{i1}^2,...,t_{ik_i^2}^2\}$, $1 \le i \le N_{I_2}$, and $T_M^2 = \{t_1,...,t_{N_{M_2}}\}$.

Step 1: Computing E and F through nested loops. For example, N_{I_1} nested loops can be used to compute E for T_I^1.

Step 2: Computing A_{ij}. A_{ij} can be calculated as follows: to join the tuples of F_i and T_D^1, scanning through the elements of F_i and for each set of **A** value, look up the tuples of T_D^1 having the same **A** value by using the index on **A**. Similarly, to join the tuples of E_i and T_D^2, scanning through the elements of E_i and for each set of **A** value, look up the tuples of T_D^2 having the same **A** value by using the index on **A**. Finally, using the nested loop approach to join the tuples of E_i and F_j.

Step 3: computing T_D, T_I, and T_M.

Step 3.1: computing T_D as follows: scanning through T_D^1 (or the smaller of T_D^1 and T_D^2) and for each set of **A** value, look up the tuples of T_D^2 having the same **A** value by using the index on **A**.

Step 3.2: computing T_I from A_{ij} in a similar way for computing E and F.

Step 3.3: computing T_M by using indices on T_D^1, T_D^2, T_M^1, and T_M^2, similar to Step 3.1.

Theorem 3.3.1: The number of pair-up operations required by Algorithm 3.3.1 grows exponentially with respect to the size of the underlying I-tables.

Theorem 3.3.2: The number of I/O block accesses required by Algorithm 3.3.1 grows exponentially with respect to the size of the underlying I-tables.

The above algorithm for evaluating natural joins of I-tables is straightforward and simple. However, this method is obviously quite inefficient as indicated by Theorem 3.3.1 and Theorem 3.3.2. The exponential complexity is not surprising at all due to the combinatorial nature of the natural joins as both E and F contain an exponential number of elements. Therefore, in general, an exponential number of elements must be enumerated by any Turing machine computing E and F.

The key to eliminating this exponential complexity is to avoid generating E and F. This can be done by comparing and concatenating T_I^1 and T_I^2, T_I^1 and T_D^2, T_I^1 and T_M^2, T_I^2 and T_D^1, and T_I^2 and T_M^1 respectively. These results are then used to derive A_{ij} directly. Therefore, the steps for forming E and F combinatorially are eliminated.

The following algorithm demonstrates the above idea and other improvements to the naive approach.

Algorithm 3.3.2: Improved algorithm for computing the natural join of I-tables T_1 and T_2.

Let $T_D^1 = \{t_1^1,...,t_{N_{D_1}}^1\}, T_I^1 = \{w_1^1,...,w_{N_{I_1}}^1\}$, where $w_i^1 = \{t_{i1}^1,...,t_{ik_i^1}^1\}$, $1 \le i \le N_{I_1}$, and $T_M^1 = \{t_1,...,t_{N_{M_1}}\}$. Let $T_D^2 = \{t_1^2,...,t_{N_{D_2}}^2\}$, $T_I^2 = \{w_1^2,...,w_{N_{I_2}}^2\}$, where $w_i^2 = \{t_{i1}^2,...,t_{ik_i^2}^2\}$, $1 \le i \le N_{I_2}$, and $T_M^2 = \{t_1,...,t_{N_{M_2}}\}$.

Step 1: Compare and concatenate T_I^1 with T_D^2 and T_M^2 respectively, compare and concatenate T_I^2 with T_D^1 and T_M^1 respectively, and compare and concatenate T_I^1 with T_I^2. The idea here is to eliminate the step for obtaining E and F with exponential number of elements with respect to the size of T_I^1 and T_I^2 and then derive A_{ij}, thus causing exponential

number of pair-up operations. This is similar to apply selection and projection operations first and then perform cartisian products. Specifically, each tuple of T_D^2 and T_M^2 is joined with T_I^1 and form new tables **ID** and **IM** respectively with the same structure as T_I^1. Similarly, each tuple of T_D^1 and T_M^1 is joined with T_I^2 and constitute new tables **DI** and **MI** respectively with the same structure as T_I^2. Furthermore, each tuple of every tuple set of T_I^2 is joined with T_I^1 to build a new table **II** with the same structure as T_I^1. The tables **ID, IM, DI, MI,** and **II** are called *disjunctive join tables* and can be obtained according to the following definition.

Definition 3.3.1: Let T_1 and T_2 be I-tables as defined above, then:

(1) **ID** = $\{ID^1, ID^2, ..., ID^{N_D^2}\}$, where

$ID^k = \{ID_{i1}^k ID_{i2}^k ..., ID_{ik_i}^k\}$, $1 \le k \le N_D^2$ and $1 \le i \le N_I^1$, and

$$ID_{ij}^k = \begin{cases} \Pi_{\mathbf{B}}(t_{ij}^1 t_k^2) & \text{if } t_{ij}^1[\mathbf{A}] = t_k^2[\mathbf{A}] \\ \varepsilon & \textit{otherwise} \end{cases}$$

where $t_{ij}^1 \in w_i^1$ of T_I^1 and $t_k^2 \in T_D^2$.

(2) **IM** = $\{IM^1, IM^2, ..., IM^{N_M^2}\}$, where

$IM^k = \{IM_{i1}^k IM_{i2}^k ..., IM_{ik_i}^k\}$, $1 \le k \le N_M^2$ and $1 \le i \le N_I^1$, and

$$IM_{ij}^k = \begin{cases} \Pi_{\mathbf{B}}(t_{ij}^1 t_k^2) & \text{if } t_{ij}^1[\mathbf{A}] = t_k^2[\mathbf{A}] \\ \varepsilon & \textit{otherwise} \end{cases}$$

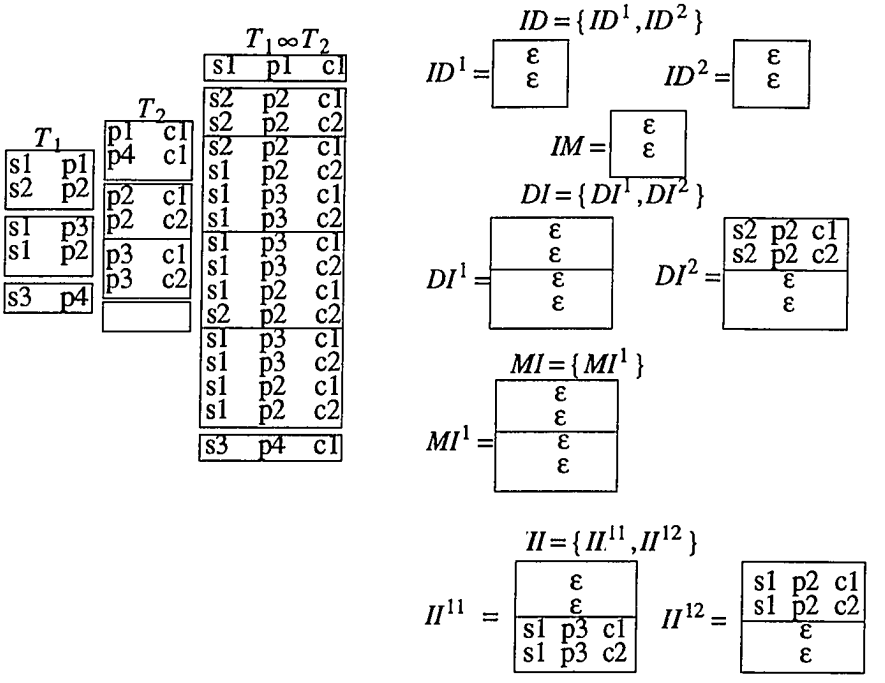
where $t_{ij}^1 \in w_i^1$ of T_I^1 and $t_k^2 \in T_M^2$.

(3) **DI** = $\{DI^1, DI^2, ..., DI^{N_D^1}\}$, where

$DI^k = \{DI_{i1}^k DI_{i2}^k ..., DI_{ik_i}^k\}$, $1 \le k \le N_D^1$ and $1 \le i \le N_I^2$, and

$$DI_{ij}^k = \begin{cases} \Pi_{\mathbf{B}}(t_k^1 t_{ij}^2) & \text{if } t_k^1[\mathbf{A}] = t_{ij}^2[\mathbf{A}] \\ \varepsilon & \textit{otherwise} \end{cases}$$

where $t_k^1 \in T_D^1$ and $t_{ij}^2 \in w_i^2$ *of* T_I^2.



$EE_1 = \emptyset$, $EE_2 = \emptyset$

$FF_1 = \{s2\ p2\ c1\}$, $FF_{2=}\{s2\ p2\ c1\}$, $FF_{3=}\{s2\ p2\ c2\}$, $FF_4 = \{s2\ p2\ c2\}$

$EF_{11=}\{s1\ p3\ c1\}$, $EF_{12} = \{s1\ p3\ c2\}$, $EF_{13=}\{s1\ p3\ c1\}$, $EF_{14=}\{s1\ p3\ c2\}$

$EF_{21=}\{s1\ p2\ c1\}$, $EF_{22} = \{s1\ p2\ c2\}$, $EF_{23=}\{s1\ p2\ c1\}$, $EF_{24=}\{s1\ p2\ c2\}$

$A_{11} = \{s2\ p2\ c1, s1\ p3\ c1\}$
$A_{12=}\{s2\ p2\ c1, s1\ p3\ c2\}$
$A_{13=}\{s2\ p2\ c2, s1\ p3\ c1\}$
$A_{14=}\{s2\ p2\ c2, s1\ p3\ c2\}$
$A_{21=}\{s2\ p2\ c1, s1\ p2\ c1\}$
$A_{22=}\{s2\ p2\ c1, s1\ p2\ c2\}$
$A_{23=}\{s2\ p2\ c2, s1\ p2\ c2\}$
$A_{24=}\{s2\ p2\ c2, s1\ p2\ c2\}$

Figure 3.4: An Example Illustrating Definition 3.3.1 and Algorithm 3.3.2

(4) **MI** = $\{MI^1, MI^2, \ldots, MI^{N_M^1}\}$, where

$MI^k = \{MI_{i1}^k MI_{i2}^k \ldots, MI_{ik_i^1}^k\}$, $1 \le k \le N_M^1$ and $1 \le i \le N_I^2$, and

$$MI_{ij}^k = \begin{cases} \Pi_{\mathbf{B}}(t_k^1 t_{ij}^2) & \text{if } t_k^1[\mathbf{A}] = t_{ij}^2[\mathbf{A}] \\ \varepsilon & \textit{otherwise} \end{cases}$$

where $t_k^1 \in T_M^1$ and $t_{ij}^2 \in w_i^2$ *of* T_I^2.

(5) **II** = $\{II^{11}, \ldots, II^{1k_1^1}, \ldots, II^{N_{I_1}1}, \ldots, II^{N_{I_1}k_{N_{I_1}}^1}\}$, where

$II^{kl} = \{II_{i1}^{kl}, II_{i2}^k \ldots, II_{ik_j^2}^k\}$, $1 \le k \le N_{I_1}$, $1 \le l \le k_k^1$, $1 \le i \le N_{I_1}$, and

$$II_{ij}^{kl} = \begin{cases} \Pi_{\mathbf{B}}(t_{ij}^1 t_{kl}^2) & \text{if } t_{ij}^1[\mathbf{A}] = t_{kl}^2[\mathbf{A}] \\ \varepsilon & \textit{otherwise} \end{cases}$$

where $t_{ij}^1 \in w_i^1$ *of* T_I^1 and $t_{kl}^2 \in T_k^2$ *of* w_k^2 *of* T_I^2.

Indices on **A** can be used to obtain the disjunctive join tables. Definition 3.3.1 is exemplified in Figure 3.4.

Step 2: Obtain A_{ij}'s: note that the disjunctive join tables are of the same structures as the indefinite components of the I-tables. Hence A_{ij} can be generated from *DI*, *ID*, and *II*, similar to generating *E* and *F*, according to the following definition:

Definition 3.3.2: Let *ID*, *DI*, and *II* be disjunctive join tables, and let

$$EE = \left\{ \{t_{1j_1}^1, t_{1j_1}^2, \ldots, t_{1j_1}^{N_{D_2}}, \cdots, t_{N_{I_1}j_{N_{I_1}}^1}, \ldots, t_{N_{I_1}j_{N_{I_1}}}^{N_{D_2}}\} \mid \right.$$
$$\left. (\forall l)(\forall m)((1 \le l \le N_{D_2} \wedge 1 \le m \le N_{I_1} \rightarrow (\exists i)(1 \le i \le k_m^1 \wedge j_m = i \wedge t_{mj_m}^l \in ID_m^l \wedge t_{mj_m}^l \ne \varepsilon)) \right\}$$

$$FF = \left\{ \{t_{1j_1}^1, t_{1j_1}^2, \ldots, t_{1j_1}^{N_{D_1}}, \cdots, t_{N_{I_2}j_{N_{I_2}}^1}, \ldots, t_{N_{I_2}j_{N_{I_2}}}^{N_{D_1}}\} \mid \right.$$
$$\left. (\forall l)(\forall m)((1 \le l \le N_{D_1} \wedge 1 \le m \le N_{I_2} \rightarrow (\exists i)(1 \le i \le k_m^2 \wedge j_m = i \wedge t_{mj_m}^l \in DI_m^l \wedge t_{mj_m}^l \ne \varepsilon)) \right\}$$

$$EF_{index} = \left\{ \{t_{1j_1}^1, \cdots, t_{N_{I_1}j_{N_I}}^1 t_{1j_1}^2, \cdots, t_{N_{I_1}j_{N_I}}^2, \ldots, t_{1j_1}^{N_{I_1}}, \cdots, t_{N_{I_1}j_{N_I}}^{N_{I_1}}\} \mid \right.$$

$$(\forall l)(\forall m)(1\le l\le N_{I_1})(1\le m\le N_{I_1}\rightarrow(\exists n)(1\le n\le k_m^{1)(i_{mj_m}^{li_l}}=II_{mj_m}^{li_l}\Big\}$$

where $1\le i_j\le k_j^1$, $1\le j\le N_{I_1}$.

and $index=(i_1-1)\times\prod_{n=2}^{N_{I_1}}k_n^1+(i_2-1)\times\prod_{n=3}^{N_{I_1}}k_n^1+\cdots+(i_{N_{I_1}-1}-1)\times k_{N_{I_1}}+i_{N_{I_1}}$.

Let the elements of EE be EE_k, $1\le i\le\prod_{n=1}^{N_{I_1}}k_n^1$, and those of FF be FF_i, $1\le i\le\prod_{n=1}^{N_{I_2}}k_n^2$. Also, let the elements of EF_i, where $1\le i\le\prod_{n=1}^{N_{I_1}}k_n^1$, be EF_{ij}, $1\le j\le i\prod_{n=1}^{N_{I_2}}k_n^2$. Then:

$A_{ij}=EE_i\cup FF_j\cup EF_{ij}$, where $1\le i\le\prod_{n=1}^{N_{I_1}}k_n^1$, and $1\le j\le\prod_{n=1}^{N_{I_2}}k_n^2$.

Definition 3.3.2 is also exemplified in Figure 3.4.

For this method, the elements of *EE*, *FF*, and *EF* must be subscripted in the same fashion, for example, from left to right. Furthermore, the order in which the elements and the members of the elements of *EE*,*FF*, and *EF* are generated from their corresponding disjunctive join tables are crucial in the sense that they have to be generated in the same sequence. The nested loop mechanism of Algorithm 3.3.1 can be used to generate *EE*, *FF*, and *EF* by adopting a uniformed looping sequence, for example, the outmost loop loops on the elements of the first tuple set and the innermost loop loops on the elements of the last tuple set. However, this mechanism is not generic and flexible because the number of nested loops varies with the size of the indefinite component of I-tables. Here we propose a general algorithm. First let us look at an example: let T_I^1 be as shown in Figure 3.5, and E_{ij} contains a tuple from w_j^1, where $1\le j\le N_{I_1}$, and $1\le i\le k_1^1*k_2^1*...*k_{N_I}^1$. Figure 3.5 shows the result of E with respect to T_I^1.

T_I^1:

t_1
t_2
t_3
t_4
t_5
t_6
t_7

E	*1*	*2*	*3*
1	t_1	t_3	t_6
2	t_1	t_3	t_7
3	t_1	t_4	t_6
4	t_1	t_4	t_7
5	t_1	t_5	t_6
6	t_1	t_5	t_7
7	t_2	t_3	t_6
8	t_2	t_3	t_7
9	t_2	t_4	t_6
10	t_2	t_4	t_7
11	t_2	t_5	t_6
12	t_2	t_5	t_7

Figure 3.5: A Sample I-table and *E*

Notice that there is a specific pattern exhibited by the above table for E: the number of times that the tuples of w_1^1 appear consecutively in E_{i1} is equal to the the number of tuples in w_2^1 multiplies the number of tuples in w_3^1, the number of times that the tuples of w_2^1 appear consecutively in E_{i2} is equal to the number of tuples in w_3^1, and the tuples in w_3^1 appear in E_{i3} consecutively once only. This pattern can be generalized as follows:

$$X = \begin{cases} \prod_{n=j+1}^{N_{I_1}} k_n^1 & \text{if } j<N_{I_1} \\ 1 & \text{if } j=N_{I_1} \end{cases}$$

where X is the number of times the tuples of w_i^1 appear in E_{ij} consecutively.

Therefore, E as well as *EE*, *FF*, *EF*, and A_{ij}'s can be generated according to the following C-like program:

$$\text{Num\_of\_Element} = \prod_{n=1}^{N_{I_1}} k_n^1;$$

$k_0^1 = 1;$

for (i=1; i<=Num_of_Element; i++)

for (j=1; j<=N_{I_1}; j++) {

$$k = ((int)(((i-1)\%(\prod_{n=j-1}^{N_{I_1}} k_n^1))/(\prod_{n=j}^{N_{I_1}} k_n^1)) + 1);$$

$E_{ij} = t_{ik}$

}

Figure 3.6: Systematic Way of Computing E and F

Step 3: computing T_D, T_I, and T_M.

Step 3.1: computing T_D as follows (supposing that $N_D^1 \leq N_D^2$): scanning through T_D^1 and for each set of **A** value, look up the tuples of T_D^2 having the same **A** value by using the index on **A**.

Step 3.2: computing T_I from A_{ij} in a similar way for computing E and F as presented in Algorithm 3.3.1.

Step 3.3: computing T_M by using indices on T_D^1, T_D^2, T_M^1, and T_M^2, similar to Step 3.1. Also, including all the elements from **IM**'s and **MI**'s.

Theorem 3.3.3: The number of pair-up operations required by Algorithm 3.3.2 grows linearly in general with respect to the size of the underlying I-tables and polynomially in the worst case with respect to the size of the underlying I-table.

Theorem 3.3.4: The number of I/O block accesses required by Algorithm 3.3.2 grows exponentially with respect to the size of the underlying I-tables.

The exponential complexity of the block accesses of Algorithm 3.3.2 is due to the fact that all the intermediate relations (e.g., *EE*, *FF*, *EF*'s, and A_{ij}'s) are actually created. However, the systematic method of creating permutations introduced in this algorithm facilitates the storage of

these temporary relations as pointers (we can view such tables as virtual relations). The rows of a virtual relation specify the composition of tuples from various components of the I-tables. If this approach is used, than the number of I/O block accesses will be reduced to a linear order of complexity with respect to the size of the underlying I-tables.

Theorem 3.3.5: The number of I/O block accesses required by Algorithm 3.3.2 grows linearly with respect to the size of the underlying I-tables, if virtual relations are used for intermediate results.

4. UPDATES AND I-TABLES

Imielinski and Lipski in [ImLi84] established a framework on the semantic meaningfulness of extended relational models and formalized precise conditions to ensure soundness and completeness of extended relational operators. Extended relational systems that satisfy these conditions were called representation systems. Three extended relational models (e.g., *Codd tables* ([Bisk81], [Codd79]), *naive tables* [ImLi84], and *conditional tables* [ImLi84] introduced for dealing with incomplete information in the form of null values were studied based on these conditions. The results indicated that, with the exception of conditional tables, none of the extended models was a representation system with respective to the primitive relational operators (viz., selection, projection, cartisian product, union, and difference) and join. Abiteboul and Grahne broadened this study to include update operations and concluded that only the conditional table possessed the ability to manage updates in a semantically correct manner [AbGr85].

The issue of updating incomplete databases, which has been neglected to a certain extent, is as important and delicate as that of querying them. As a matter of fact, it has been pointed out in [AbGr85] that these two aspects are intimately related as the ability to answer queries assumes the capability to enter incomplete information into databases, and hopefully to remove uncertainties as the result of updates [Win86a]. Therefore, it is fundamental to understand the impact of updates on incomplete information. [AbGr85], [Chol88], [FaUV83], [MaWa87], [Reit88], [Win86a], [Win86b], [Win86c], and [Wins90] are among the works which investigate the subject of updates.

It has been proven that I-tables are able to correctly manage all the primitive relational operators and join in a semantically correct manner ([LiS90b], [LiZh91]). In this chapter, we will further our exploration by extending update operations to I-tables and then examining the

ability of I-tables for managing update operations in a semantically correct way. Specifically, The update operations of insertion and deletion are extended to I-tables. Both the syntactic and semantic definitions of the extended update operations are presented and their correctness, or the equivalence of the syntactic and semantic definitions, are shown. Properties of the extended update operations are also established and exhibited.

The work presented in this chapter is related to that of Abiteboul and Grahne [AbGr85]: both of the works extend the framework of the semantic meaningfulness of relational operators to update operations and base the semantics of updates on the models, or information contents associated with the underlying first-order theory. The main difference, however, lies in the information contents and the corresponding extended relational models. The extended relational models investigated in [AbGr85] are several versions of relational tables with null values. Their corresponding information contents are defined to be the set of all the possible states represented by the underlying extended models. On the other hand, I-tables, the extended relational model, or underlying first-order theory examined in this chapter, models general forms of indefinite and maybe information. Its corresponding information content consists of a set of states which corresponds to the **minimal models** of the underlying first-order theory represented by I-tables. The quest, with respect to I-tables, then encompasses the following: (1) need to determine if it is possible to use the minimal models (instead of nonminimal models) as the basis for defining the semantics of the update operations, and (2) need to extend updates for general disjunctive information of the form $P_1(t_1) \vee \cdots \vee P_n(t_n)$.

The discussion in the remainder of this chapter divides into two general areas. One area deals with the semantics of the update operations, which is the plan of Section 4.1 and Section 4.2. Section 4.1 identifies the update operations on I-tables and Section 4.2 discusses the associated semantics of the update operations identified in Section 4.1. Section 4.3 deals with the syntactic aspects of the update operations. Specifically, the extended update operations are

formally defined on I-tables. The correctness and associated properties of the extended update operations are established in Section 4.4.

4.1 Updates on I-tables

Suppose for the discussion follows that T is an I-table over a relational scheme R, and t, t_1, t_2, ..., and t_n are tuples over R. Since I-tables encompass three components, the definite component, the indefinite component, and the maybe component, insertion operations on I-tables must be able to handle the incorporation of new members into all the three components. Furthermore, the indefinite component is a collection of tuple sets, one should also be able to insert a new tuple into a tuple set of the indefinite component. Hence, there are four types of insertions on I-tables:

(1) definite insertion: $\mathbf{INS}_D^{\Gamma} < t > (T)$ - inserting tuple t into T_D,

(2) indefinite insertion (tuple set): $\mathbf{INS}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (T)$, where $n \geq 2$ - inserting tuple set $\{t_1, t_2, \cdots, t_n\}$ into T_I,

(3) indefinite insertion (tuple): $\mathbf{INS}_I^{\Gamma} < t, \{t_1, t_2, \cdots, t_n\} > (T)$, where $n \geq 2$ - inserting tuple t into the tuple set $\{t_1, t_2, \cdots, t_n\}$ of T_I, and

(4) maybe insertion: $\mathbf{INS}_M^{\Gamma} < t > (T)$ - inserting tuple t into T_M .

Figure 4.2 - Figure 4.4 gives examples of the insertion operations identified above.

The deletion operations on I-tables can be extended symmetrically to the extended insertion operations as follows:

(1) definite deletion: $\mathbf{DEL}_D^{\Gamma} < t > (T)$ - deleting tuple t from T_D.

(2) indefinite deletion (tuple set): $\mathbf{DEL}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (T)$, where $n \geq 2$ - deleting tuple set $\{t_1, t_2, \cdots t_n\}$ from T_I.

(3) indefinite deletion (tuple): $\mathbf{DEL}_I^{\Gamma} < t_i, \{t_1, t_2, \cdots, t_i, \cdots, t_n\} > (T), n \geq 3$ - deleting tuple t_i from the tuple set $\{t_1, t_2, \cdots, t_i, \cdots, t_n\}$ of T_I, and

(4) maybe deletion: $\mathbf{DEL}_M^{\Gamma} < t > (T)$ - deleting tuple t from T_M.

Figure 4.5 - Figure 4.8 gives examples of the deletion operations identified above.

Modification operations is not formalized since they can be expressed as a sequence of deletions and insertions.

4.2 Semantics of the Extended Update Operations

First, the semantics, or information content of I-tables is extended to support the extended update operations presented in the previous section. This extension of the semantic definition is necessary in order to manipulate the extended update operations in a semantically correct manner, which is the most important goal to be achieved. The semantics of I-tables is extended as follows:

Definition 4.2.1: Let $R = <A_1, \ldots, A_k>$ with domain D_i for each attribute name A_i, $1 \le i \le k$ be a relational scheme and T be a reduced I-table over R, then

$\Gamma_R = \{T | T$: *I-table over R* $\}$, and

$\Sigma_R = \{<U,v> | U$: *set of relations over R*, v: *relation over R*$\}$.

The information content of the I-table T is a mapping *REP* from Γ_R to Σ_R such that:

$REP(T) = <REDUCEREP(<MM,M>(T)), OCCUR>$, where

(1) $<MM,M>$ is a mapping from Γ_R to Σ_R such that:

$<MM,M>(T) = <MM(T), M(T)>$, where

$T = <T_D, T_I, T_M>$ with $T_I = \{w_1, \ldots, w_n\}$, and

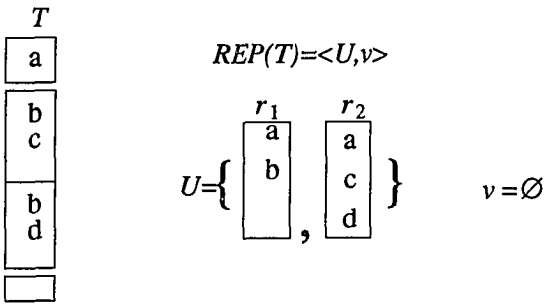
$MM(T) = \{T_D \cup \{t_1, \ldots, t_n\} | (\forall i)(1 \le i \le n \rightarrow t_i \in w_i)\}$, and

$M(T) = T_M$.

(2) *REDUCEREP* is a mapping from Σ_R to Σ_R such that:

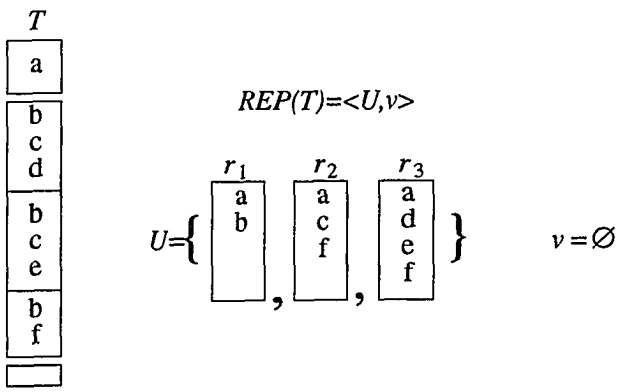
$REDUCEREP(<U,v>) = <U^o, v^o>$, where

$U^o = \{r | (r \in U \wedge \neg(\exists r_1)(r_1 \in U \wedge r_1 \subset r)\}$, and



$OCCUR(r_1,a)=1$ $OCCUR(r_1,b)=2$

$OCCUR(r_2,a)=1$ $OCCUR(r_2,c)=1$ $OCCUR(r_2,d)=1$



$OCCUR(r_1,a)=1$ $OCCUR(r_1,b)=3$

$OCCUR(r_2,a)=1$ $OCCUR(r_2,c)=2$ $OCCUR(r_2,f)=1$

$OCCUR(r_3,a)=1$ $OCCUR(r_3,d)=1$ $OCCUR(r_3,e)=1$

$OCCUR(r_3,f)=1$

Figure 4.1: The Extended Semantics of I-tables

$$v^o = \{t \mid (t \in v \vee (\exists r_1)(\exists r_2)(r_1 \in U \wedge r_2 \in U \wedge r_1 \subset r_2 \wedge t \in r_2 - r_1) \wedge \neg(\exists r)(r \in U^o \wedge t \in r))\}.$$

(3) *OCCUR* is a mapping from $\Sigma_R \times (D_1 \times \cdots \times D_k)$ to **N** (the set of natural numbers). $OCCUR(r,t)$ denotes the number of times the tuple t occurs in the definite relation r of U. Specifically, $OCCUR(\{T_D \cup \{t_1, \ldots, t_n\}, t)$, where $t_i \in w_i$, $1 \leq i \leq n$, can be computed as follows: $OCCUR(\{T_D \cup \{t_1, \ldots, t_n\}, t) = 1$ for each tuple t of T_D and $OCCUR(\{T_D \cup \{t_1, \ldots, t_n\}, t_i) = OCCUR(\{T_D \cup \{t_1, \ldots, t_n\}, t_i) + 1$ for each tuple t_i of w_i, $1 \leq i \leq n$. It is assumed here that $OCCUR(\{T_D \cup \{t_1, \ldots, t_n\}, t_i)$ is initialized to zero for each t_i of w_i, $1 \leq i \leq n$.

Note that the meaning associated with $MM(T)$, $M(T)$, U^o, and v_o is still the same as the original definition. The only difference here is that a new function *OCCUR* is added which is used to determine the actual number of times a tuple appears in a possible real world true combination. For example, the tuple b occurs twice in the definite relation $\{a,b\}$ as shown in Figure 4.1.

The notion of correctness can still be expressed by Figure 3.2. However, under the extended semantics, $< U_1, v_1 > = < U_2, v_2 >$ if $U_1 = U_2$ and $v_1 = v_2$, where $U_1 = U_2$ if they have the same members (i.e., every relation of U_1 is a relation of U_2 and every relation of U_2 is also a relation of U_1). Further, for any r_1 of U_1 and r_2 of U_2, $r_1 = r_2$ implies that $OCCUR(r_1,t)$ must be equal to $OCCUR(r_2,t)$ for each t of r_1 (or r_2).

Let T be an I-table and $< U, v > = REP(T)$, the information content of the I-table T. U then is the set of all possible minimal definite relations represented by T and of which at least one is the real world truth, yet it is not known which one or ones are true, and v is the set of all maybe tuples of T. According to this underlying semantics, or information content, of I-tables, any operator f on I-tables should be interpreted as if f is applied to all the possible states of the underlying incomplete database. This interpretation is used as the foundation for the semantics

of the extended update operations on I-tables.

First let us consider definite insertions, inserting a tuple t which is known to be true into T. Based on the underlying semantics, t should be inserted into every possible state of $POSS(U)$ to guarantee that it is in the true state or states of the real world. Furthermore, $OCCUR(r \cup t, t)$ should be set to 1 for each r of $POSS(U)$.

Attempting to extract the underlying semantics for the insertion of an indefinite tuple set $W = \{t_1, \ldots, t_n\}$, where $n \geq 2$, the following observation and analogy may be made. The tuple set W to be inserted may be viewed as a first-order theory and corresponding to W is a collection of relations W_m which is the model associated with W (e.g., the information content of W). Therefore, it is equivalent to insert a set of relations W_m into $POSS(U)$. In terms of logic, this type of insertion can be viewed as a logic "and" of two disjunctions. One of the disjunction is $POSS(U)$ and the other is W_m. In light of the distributivity of $\wedge$ over $\vee$, this implies the insertion of each definite relation into each of its corresponding possible state of $POSS(U)$. In this case, $OCCUR$ should be calculated as follows: let $r_1 \in POSS(U)$ and $r_2 \in W_m$, then $OCCUR(r_1 \cup r_2, t)$ should be increased by 1 if t is in both r_1 and r_2 whereas $OCCUR(r_1 \cup r_2, t)$ should be set to 1 if t belongs to r_2 only.

Compounding the analysis is the insertion of a tuple t into a tuple set $\{t_1, \cdots t_n\}$ of an I-table T. This type of insertion amplifies the degree of uncertainty. This is true because the combination of the possibility that t is true has to be taken into account on top of all the possibilities of valid combinations of $\{t_1, \ldots, t_n\}$ being true. The semantics of this type of insertions is best understood when placed in perspective relative to that of the insertion and deletion of tuple sets. That is, it is equivalent to inserting the tuple set $\{t, t_1, \cdots t_n\}$ into $T - \{t_1, \ldots, t_n\}$.

The other component of $REP(T)$, v, is a single relation which contains the tuples that may be true in the real world. Thus maybe insertions, inserting a tuple t into the maybe component of T,

are similar to insertions of conventional relations and should be interpreted as inserting t into v.

The semantics of the deletion operations mirror that of the insertion operations. Based on the underlying semantics, definite deletion (i.e., deleting a tuple t which is known to be true from T) should be interpreted as follows: if t belongs to all the possible state of $POSS(U)$ then t should be deleted from every possible state of $POSS(U)$ to guarantee that it is not always in the real world true state or states.

Attempting similarly to extract the underlying semantics for the deletion of an indefinite tuple set $W = \{t_1,...,t_n\}$, where $n \geq 2$, the following observation and analogy can also be made. The tuple set W to be deleted may be viewed as a first-order theory and corresponding to W is a collection of relations W_m which is the model associated with W (e.g., the information content of W). Therefore, it is equivalent to delete a set of relations W_m into $POSS(U)$. In terms of logic, this type of deletion can be viewed as a logic "difference" of two disjunctions. One of the disjunction is $POSS(U)$ and the other is W_m. Therefore, each definite relation of W_m should be deleted from each of its corresponding possible state of $POSS(U)$ as follows: let $r_1 \in POSS(U)$ and $r_2 \in W_m$, then $OCCUR(r_1 - r_2, t)$ remains unchanged if t belongs to r_1 only and t remains in $r_1 - r_2$, $OCCUR(r_1 - r_2, t)$ should be decreased by 1 if t is in both r_1 and r_2 and $OCCUR(r_1, t) > 1$ and t remains in the result of $r_1 - r_2$, $OCCUR(r_1 \cup r_2, t)$ should be set to 0 if t is in both r_1 and r_2 and $OCCUR(r_1, t) = 1$ and t is removed from $r_1 - r_2$, and $OCCUR(r_1 - r_2, t)$ does not exist if t belongs to r_2 only.

The deletion of a tuple t from a tuple set $\{t, t_1, \cdots t_n\}$ of an I-table T. This type of insertion decreases the degree of uncertainty. This is true because the combination of the possibility that t is true is no longer valid. Similarly, The semantics of this type of insertions is best understood when placed in perspective relative to that of the insertion and deletion of tuple sets. That is, it is equivalent to inserting the tuple set $\{t_1, \cdots t_n\}$ into $T - \{t, t_1, \ldots, t_n\}$.

Finally, the semantics of maybe deletion, deleting a maybe tuple t from the maybe

component of T, is simply the deletion of t from v.

Note that $U_1 \subseteq U_2$ implies that $\mathbf{INS}_D^{\Sigma}(<t>(<U_1,v>) \subseteq \mathbf{INS}_D^{\Sigma}(<t>(<U_2,v>)$, $\mathbf{INS}_I^{\Sigma}(<\{t_1,t_2,\cdots,t_n\}>)(<U_1,v>) \subseteq \mathbf{INS}_I^{\Sigma}(<\{t_1,t_2,\cdots,t_n\})(<U_2,v>)$, and $\mathbf{INS}_{I_t}^{\Sigma}(<\{t_1,t_2,\cdots,t_n\}>)(<U_1,v>) \subseteq \mathbf{INS}_{I_t}^{\Sigma}(<\{t_1,t_2,\cdots,t_n\})(<U_2,v>)$. Also, $v_1 \subseteq v_2$ implies that $\mathbf{INS}_M^{\Sigma}(<t>(<U,v_1>) \subseteq \mathbf{INS}_M^{\Sigma}(<t>(<U,v_2>)$. These properties also applies to the various deletion operations.

Based on the above discussion and the definition of *REDUCEREP*, the fact $U \subseteq POSS(U)$, the semantics of the extended insertion operations are formally defined as follows.

Definition 4.2.2: Let R be a relational scheme, T be a reduced I-table over R. For simplicity, assume that $OCCUR(r,t)=0$ if t is not a member of r. Then

(1) $\mathbf{INS}_D^{\Sigma}<\{t\}>(<U,v>)=REDUCEREP<U',v'>$, where

$U' = \{r \mid (\exists r_1)(r_1 \in U \wedge r = r_1 \cup t) \wedge OCCUR(r,t)=1\}$, and

$v'=v$.

(2) $\mathbf{INS}_I^{\Sigma}<\{t_1,\cdots,t_n\}>(<U,v>)=REDUCEREP<U',v'>$, where

$$U' = \{r \mid (\exists r_1)(\exists t)((r_1 \in U) \wedge (t \in \{t_1,\ldots,t_n\} \wedge (r = r_1 \cup t) \wedge OCCUR(r,t)=OCCUR(r_1,t)+1\}$$

$v'=v$.

(3) $\mathbf{INS}_M^{\Sigma}<REP(\{t\})>(<U,v>)=REDUCEREP<U',v'>$, where

$U' = U$ and $v'=v \cup \{t\}$.

(4) $\mathbf{DEL}_D^{\Sigma}<REP(\{t\})>(<U,v>)=REDUCEREP<U',v'>$, where

$$U' = \{r \mid (\exists r_1)(r_1 \in U \wedge \left[(r = r_1 - t \wedge t \in \bigcap_{r \in U} r \wedge OCCUR(r,t)=0) \vee (r = r_1 \wedge t \notin \bigcap_{r \in U} r)\right]\}, \text{ and}$$

$v'=v$.

(5)$\mathbf{DEL}_I^{\Sigma}<\{t_1,\cdots,t_n\}>(<U,v>)=REDUCEREP<U',v'>$, where

$U' = \{r \mid (\exists r_1)(\exists t)((r_1 \in U) \wedge (t \in \{t_1,\ldots,t_n\} \wedge$

$((t \in r_1 \wedge OCCUR(r_1,t) > 1 \wedge r=r_1 \wedge OCCUR(r,t)=OCCUR(r_1,t)-1) \vee$

$((t \in r_1 \wedge OCCUR(r_1,t)=1 \wedge r=r_1-t \wedge OCCUR(r,t)=0) \vee$

$((t \notin r_1 \wedge r=r_1))\}$, and

$v'=M(T)$.

(6) $\mathbf{DEL}_M^{\Sigma}<REP(\{t\})>(<U,v>)=REDUCEREP<U',v'>$, where

$U' = U$ and $v'=v-\{t\}$.

(7)$\mathbf{INS}_I^{\Sigma}<\{t\},\{t_1,\cdots,t_n\}>(<U,v>))=$

$\mathbf{INS}_I^{\Sigma}<\{t,t_1,\cdots,t_n\}>(\mathbf{DEL}_I^{\Sigma}<\{t_1,\cdots,t_n\}>(<U,v>))$.

(8)$\mathbf{DEL}_I^{\Sigma}<\{t_i\},\{t_1,\cdots,t_i,\ldots,t_n\}>(<U,v>))=$

$\mathbf{INS}_I^{\Sigma}<\{t_1,\cdots,t_{(i-1)},t_{(i+1)1},\ldots,t_n\}>(\mathbf{DEL}_I^{\Sigma}<\{t_1,\cdots,t_i,\ldots,t_n\}>(<U,v>))$.

4.3 Extended Update Operations on I-tables

The extended update operations are now defined on I-tables.

Definition 4.3.1: Let T be an I-table over relational scheme R. Then:

(1) $\mathbf{INS}_D^{\Gamma}<t>(T)=REDUCE(T')$, where $T'_D=T_D \cup \{t\}$, $T'_I=T_I$, and $T'_M=T_M$,

(2) $\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T)=REDUCE(T')$, where

$T'_D=T_D$, $T'_I=T_I \cup \{t_1,t_2,\cdots,t_n\}$, and $T'_M=T_M$,

(3) $\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_n\}>(T)=REDUCE(T')$, where

$T'_D=T_D, T'_I=(T_I-\{t_1,t_2,\cdots,t_n\}) \cup \{t,t_1,t_2,\cdots,t_n\}$, and

$T'_M=T_M$, and

(4) $\mathbf{INS}_M^{\Gamma}<t>(T)=REDUCE(T')$, where $T'_D=T_D$, $T'_I=T_I$, and $T'_M=T_M \cup \{t\}$.

Definition 4.4.2 : Let T be an I-table over relational scheme R. Then:

(1) $\mathbf{DEL}_D^{\Gamma} < t > (T) = T'$, where $T_D' = T_D - \{t\}$, $T_I' = T_I$, and $T_M' = T_M$,

(2) $\mathbf{DEL}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (T) = T'$, where $T_D' = T_D$, $T_I' = T_I - \{t_1, t_2, \cdots, t_n\}$, and $T_M' = T_M$,

(3) $\mathbf{DEL}_I^{\Gamma} < t_i, \{t_1, t_2, \cdots, t_i, \cdots, t_n\} > (T) = T'$, where $T_D' = T_D$, $T_I' = (T_I - \{t_1, t_2, \cdots, t_i, \cdots, t_n\}) \cup \{t_1, t_2, \cdots, t_{i-1}, t_{i+1}, \cdots, t_n\}$, and $T_M' = T_M$, and

(4) $\mathbf{DEL}_M^{\Gamma} < t > (T) = T'$, where $T_D' = T_D$, $T_I' = T_I$, and $T_M' = T_M - \{t\}$.

4.4 Results and Properties

This section contains two basic results of this chapter. Firstly, the semantic correctness of the extended update operations of insertion and deletion is established. Secondly, some properties associated with the extended insertion and deletion operators are determined.

Theorem 4.4.1: Let R be a relational scheme and T be a reduced I-table over R, then:

(1) $\mathbf{INS}_D^{\Sigma} < t > (REP(T)) = REP(\mathbf{INS}_D^{\Gamma} < t > (T))$,

(2) $\mathbf{INS}_I^{\Sigma} < \{t_1, t_2, \cdots, t_n\} > (REP(T)) = REP(\mathbf{INS}_{I_{is}}^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (T))$,

(3) $\mathbf{INS}_M^{\Sigma} < t > (REP(T)) = REP(\mathbf{INS}_M^{\Gamma} < t > (T))$,

(4) $\mathbf{DEL}_D^{\Sigma} < t > (REP(T)) = REP(\mathbf{DEL}_D^{\Gamma} < t > (T))$,

(5) $\mathbf{DEL}_I^{\Sigma} < \{t_1, t_2, \cdots, t_n\} > (REP(T)) = REP(\mathbf{DEL}_{I_{is}}^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (T))$,

(6) $\mathbf{DEL}_M^{\Sigma} < t > (REP(T)) = REP(\mathbf{DEL}_M^{\Gamma} < t > (T))$,

(7) $\mathbf{INS}_I^{\Sigma} < t, \{t_1, t_2, \cdots, t_n\} > (< U, v >) = REP(\mathbf{INS}_{I_t}^{\Gamma} < t, \{t_1, t_2, \cdots, t_n\} > (T))$,

(8) $\mathbf{DEL}_I^{\Sigma} < t_i, \{t_1, \ldots, t_i, \cdots, t_n\} > (< U, v >) =$

$REP(\mathbf{DEL}_{I_t}^{\Gamma} < t_i, \{t_1, \ldots, t_i, \cdots, t_n\} > (T))$.

Theorem 4.4.1, except (3) and (6) which are straightforward, is illustrated in Figure 4.2, Figure 4.3, Figure 4.4, Figure 4.5, Figure 4.6, Figure 4.7, and Figure 4.8.

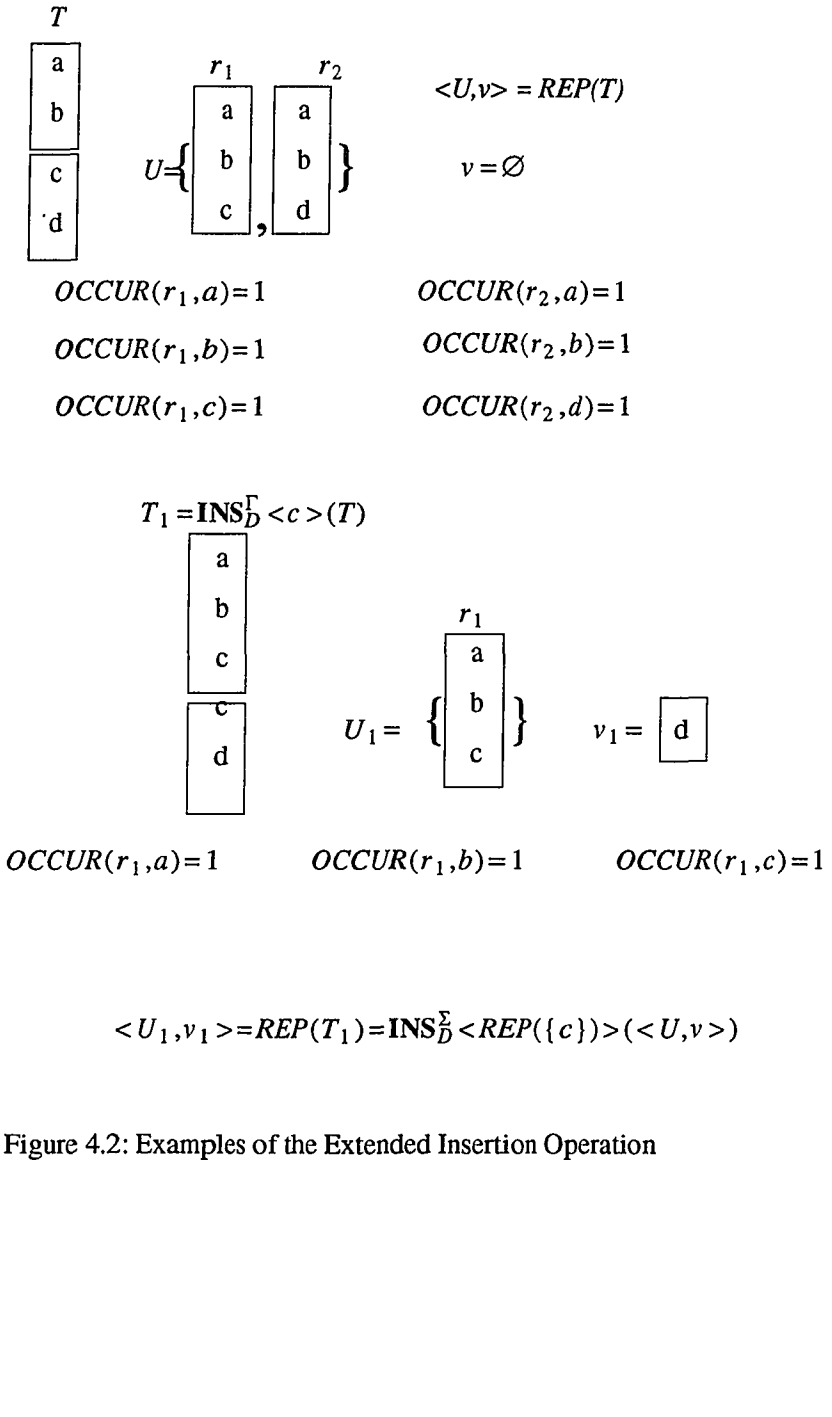


Figure 4.2: Examples of the Extended Insertion Operation

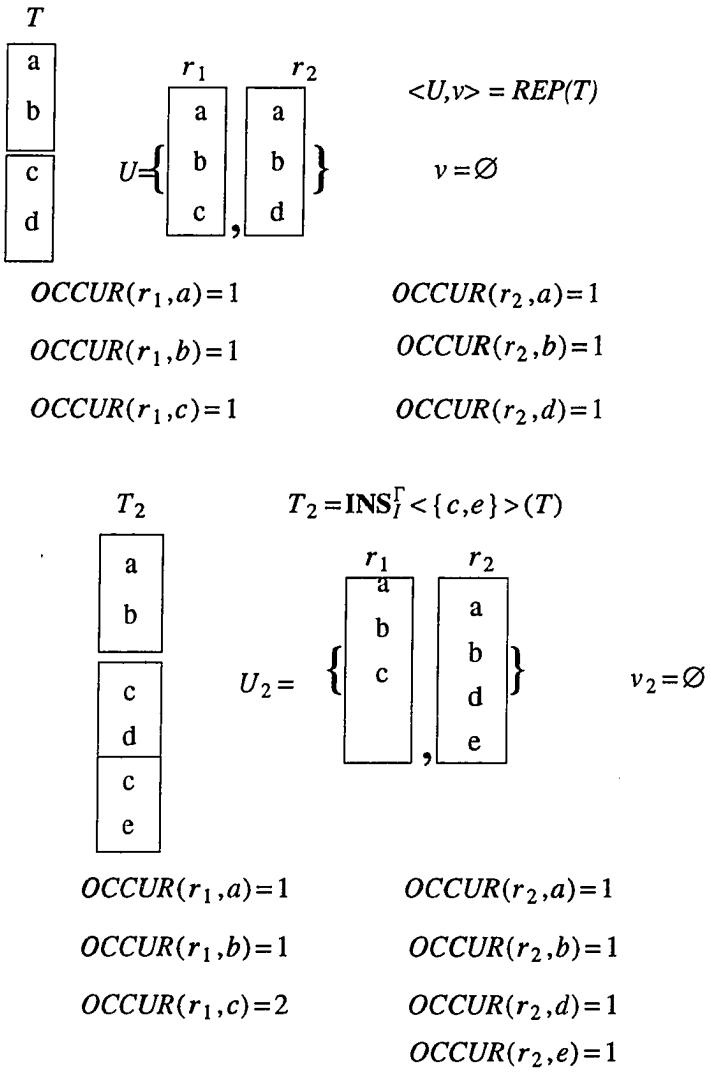
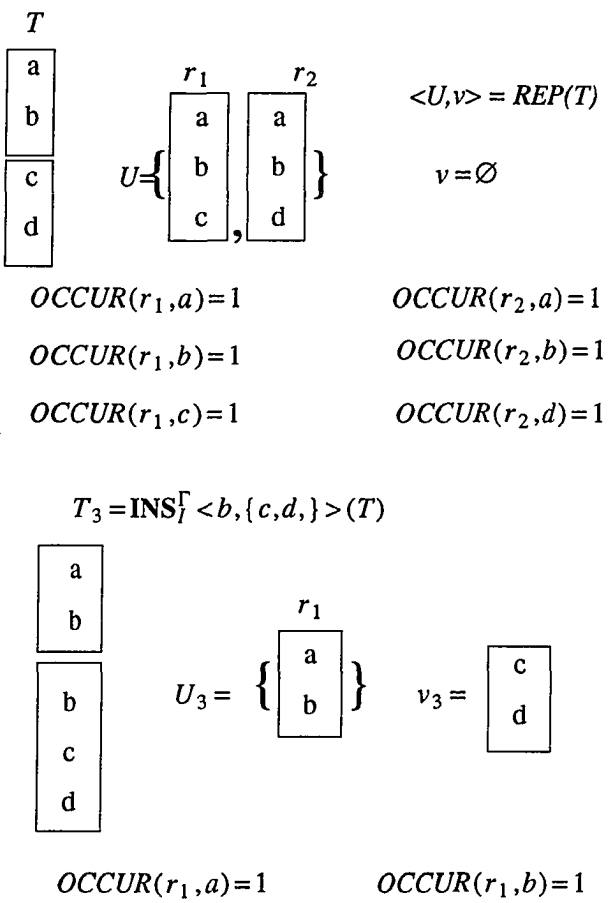
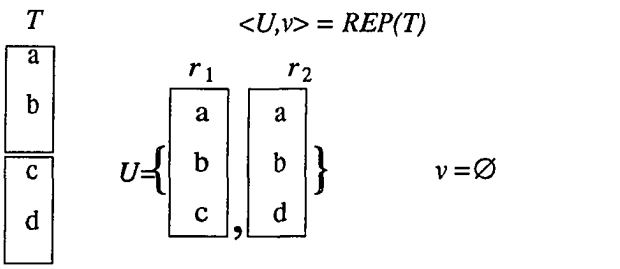


Figure 4.3: Examples of the Extended Insertion Operation



$$< U_3, v_3 > = REP(T_3) = \mathbf{INS}_I^{\Sigma} < REP(\{b\}), REP(\{\{c,d\}\}) > (< U, v >)$$

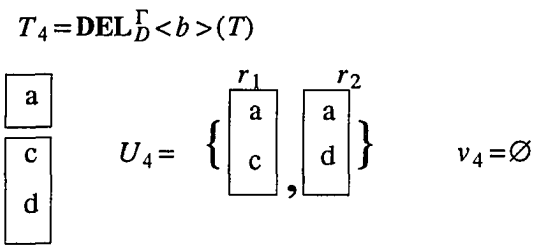
Figure 4.4: Examples of the Extended Insertion Operation



$OCCUR(r_1,a)=1$ $OCCUR(r_2,a)=1$

$OCCUR(r_1,b)=1$ $OCCUR(r_2,b)=1$

$OCCUR(r_1,c)=1$ $OCCUR(r_2,d)=1$

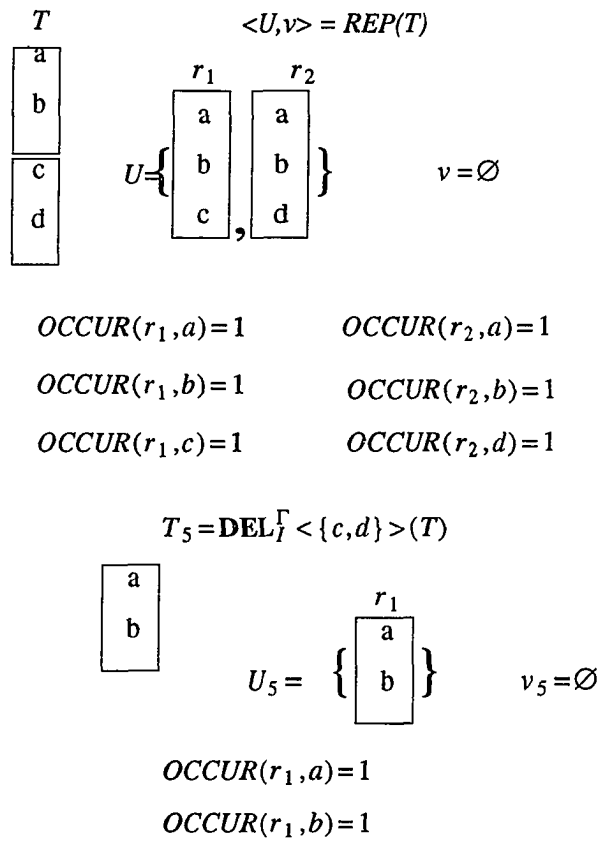


$OCCUR(r_1,a)=1$ $OCCUR(r_2,a)=1$

$OCCUR(r_1,c)=1$ $OCCUR(r_2,d)=1$

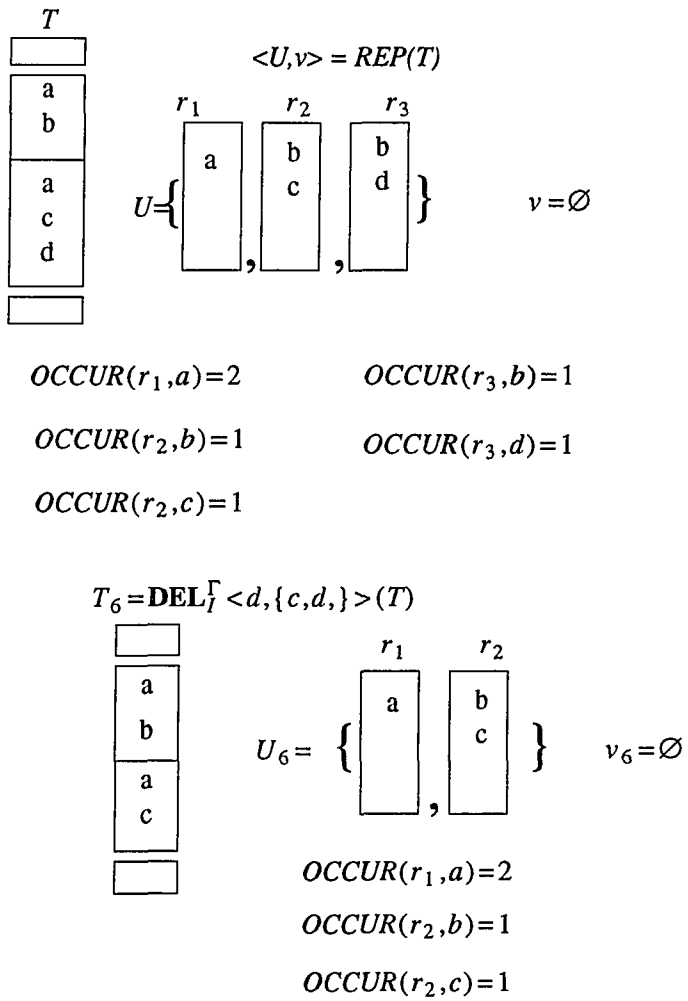
$$<U_4,v_4>=REP(T_4)=\mathbf{DEL}_D^{\Sigma}<REP(\{b\})>(<U,v>)$$

Figure 4.5: Examples of the Extended Deletion Operation



$$< U_5, v_5 > = REP(T_5) = \mathbf{DEL}_I^{\Sigma} < REP(\{\{c,d\}\}) > (< U, v >)$$

Figure 4.6: Examples of the Extended Deletion Operation



$$<U_6,v_6>=REP(T_6)=\mathbf{DEL}_I^{\Sigma}<REP(\{d\},REP(\{\{c,d\}\}>(<U,v>)$$

Figure 4.7: Examples of the Extended Deletion Operation

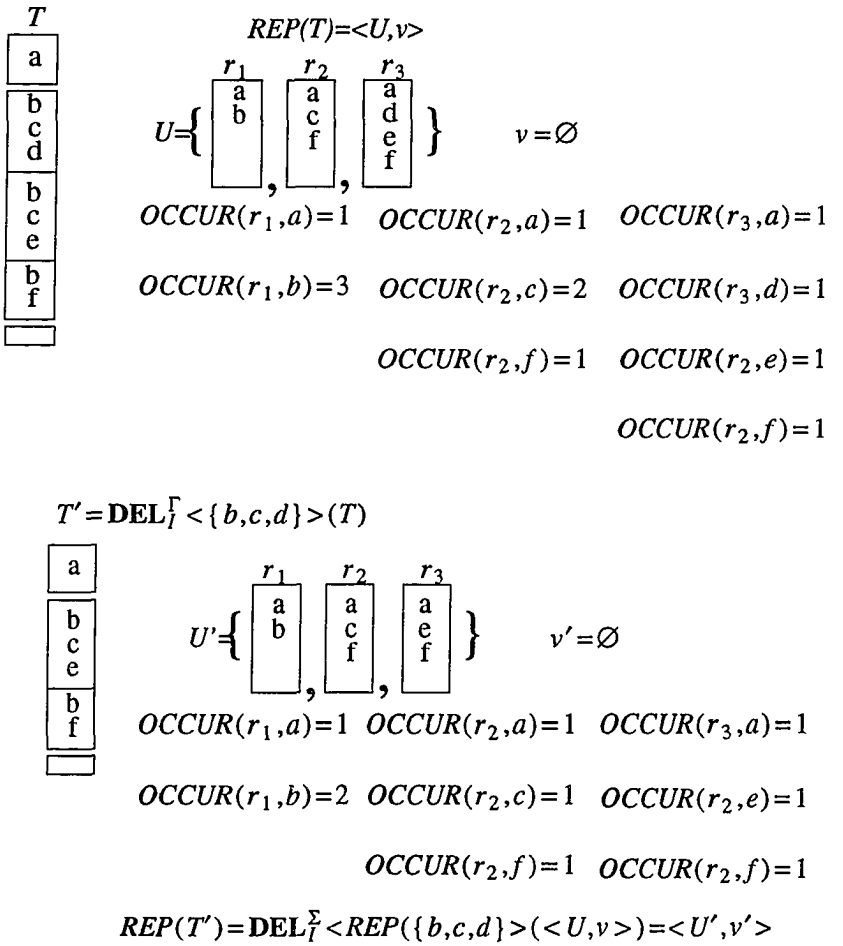


Figure 4.8: Examples of the Extended Deletion Operation

For conventional relational databases, inserting a tuple into a relation and then deleting it yields the original relation. This property also holds if deletion is performed first and followed by insertion. However, the update operations of insertion and deletion on I-tables exhibit different behaviors as established in the following theorem.

Theorem 4.4.2: Let R be a relational scheme and T be a reduced I-table over R, then:

(1) $\mathbf{DEL}_D^{\Gamma}<t>(\mathbf{INS}_D^{\Gamma}<t>(T))\neq T$

(2) $\mathbf{INS}_D^{\Gamma}<t>(\mathbf{DEL}_D^{\Gamma}<t>(T))=T$

(3) $\mathbf{DEL}_D^{\Gamma}<t>(\mathbf{INS}_D^{\Gamma}<t>(T))\neq\mathbf{INS}_D^{\Gamma}<t>(\mathbf{DEL}_D^{\Gamma}<t>(T))$

(4) $\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))\neq T$

(5) $\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))=T$

(6) $\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))$

$\neq\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))$

(7) $\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_n\}>(\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_n\}>(T))\neq T$

(8) $\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t_{m+1},\cdots,t_n\}>$

$(\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t,t_{m+1},\cdots,t_n\}>(T))=T$

(9) $\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t_{m+1},\cdots,t_n\}>$

$(\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t,t_{m+1},\cdots,t_n\}>(T))\neq$

$\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t_{m+1},\cdots,t_n\}>$

$(\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t,t_{m+1},\cdots,t_n\}>(T))$

(10) $\mathbf{DEL}_M^{\Gamma}<t>(\mathbf{INS}_M^{\Gamma}<t>(T))=\mathbf{INS}_M^{\Gamma}<t>(\mathbf{DEL}_M^{\Gamma}<t>(T))=T$

5. UPDATES AND M-TABLES

This chapter furthers the study presented in Chapter 4 by extending update operations to M-tables in a semantically correct manner. Both the syntactic and the semantic definitions of the extended update operations are formalized. The ability of M-tables for handling update operations in a semantically correct manner, that is the equivalence of the syntactic and the semantic definitions, is claimed. The results obtained in this chapter and previous work indicate that the M-table model constitutes a representation system with respect to the primitive relational operators and the update operations defined herein. Properties of the extended update operations are also presented.

In the remainder of this chapter, we first present in Section 5.1 and Section 5.2 some background knowledge regarding M-tables, including the information content and the concept of redundancies in M-tables. The plan for Section 5.3 is as follows: Section 5.3.1 establishes the notion of correctness of the update operations. Update operations on M-tables are then identified in Section 5.3.2, followed by the discussion of their associated semantics in Section 5.3.3. Section 5.3.4 formally defines the extended update operations on M-tables. Finally, the correctness and associated properties of these update operations are established in Section 5.4.

5.1 M-tables and Their Associated Semantics

Informally, an *M-table* T over a mixed relational, or M-relational scheme $MR = <R_1, R_2, \cdots, R_k>$ is a group of k mixed relations, or M-relations $<r_1, \ldots, r_k>$, where r_i is a relation over $R_i, 1 \leq i \leq k$, respectively. It consists of two components: the *sure component* T_{sure} and the *maybe component* T_{maybe}. The *sure component* of an M-table is a set of mixed tuple, or M-tuple sets of arities greater than or equal to 1. A mixed tuple, or M-tuple is of the form $<t_1, t_2, \cdots, t_k>$, where t_i is a tuple over relational scheme $R_i, 1 \leq i \leq k$, and each M-tuple

set is known to be true. The *maybe component* consists of a set of mixed tuples, each of which may be true.

Two dimensions of indefinite information can be represented in an M-table: vertical and horizontal. Vertical indefinite information refers to disjunctive information within a single relation and are represented via M-tuple sets of arity two or more of the sure component in M-tables. On the other hand, horizontal indefinite information alludes to uncertainties among different relations and are represented via M-tuple sets with at least two tuples over different relational schemes.

FATHER	SON	MOTHER	SON	FATHER	DAUGHTER	MOTHER	DAUGHTER
John	Joe						
Chris	Jack	Chris	Jack				
		Mary Nancy	Sam Sam			Mary Nancy	Sam Sam
				Steve Jack	Linda Linda		
Leslie	Pat	Leslie	Pat	Leslie	Pat	Leslie	Pat
Ed	Mike						
Kim	Mark	Kim	Mark				

Figure 5.1: PARENT-CHILDREN M-relation

By way of example, Figure 5.1 illustrates an M-table *PARENT-CHILDREN* over M-relational scheme *<FATHER-SON, MOTHER-SON, FATHER-DAUGHTER, MOTHER-DAUGHTER>*.

$PARENT-CHILDREN_{sure}$ represents the following information, or ground formulas:

GF1. FATHER-SON(John, Joe)

GF2. FATHER-SON(Chris, Jack) $\vee$ MOTHER-SON(Chris, Jack)

GF3. MOTHER-SON(Mary, Sam) $\vee$ MOTHER-DAUGHTER(Mary, Sam) $\vee$
MOTHER-SON(Mary, Sam) $\vee$ MOTHER-DAUGHTER(Mary, Sam)

GF4. FATHER-DAUGHTER(Steve, Linda) ∨ FATHER-DAUGHTER(Jack, Linda)

GF5. FATHER-SON(Leslie, Pat) ∨ MOTHER-SON(Leslie, Pat) ∨

FATHER-DAUGHTER(Leslie, Pat) ∨ MOTHER-DAUGHTER(Leslie, Pat)

Each of the above five ground formulas are true, while it is not known which ground clause or clauses are true in GF2, GF3, GF4, and GF5. Moreover, the M-tuple sets corresponds to GF2 is an example of horizontal disjunctions across the relations $FATHER-SON$ and $MOTHER-SON$ while the one associated with GF4 signifies vertical disjunctions within the relation $FATHER-DAUGHTER$. Yet GF3 is derived from the M-tuple set incorporating both vertical and horizontal indefinite information.

$PARENT-CHILDREN_{maybe}$ represents the following ground atomic formulas:

GAF1. FATHER-SON(Ed, Mark)

GAF2. FATHER-SON(Kim, Mark) ∨ MATHER-SON(Kim, Mark)

However, these ground atomic formulas may or may not be true.

Formally, an M-table T over an M-relational scheme $MR=<R_1, R_2, \ldots, R_k>$ with attribute names $A_i^1, \ldots\ldots A_i^{n_i}$ and associated domains D_i^j, where $1 \le i \le k$ and $1 \le j \le n_i$, is defined as $T = <T_{sure}, T_{maybe}>$, where

$$T_{sure} \subseteq \{<t_1, \ldots, t_k> \mid (\forall i)(1 \le i \le k \rightarrow t_i \in 2^{D_i^1 \times \cdots \times D_i^{n_i}}) \wedge (\exists i)(1 \le i \le k \wedge t_i \neq \varnothing)\}$$

$$T_{maybe} \subseteq \{<r_1, \cdots, r_k> \mid (\forall)(1 \le i \le k \rightarrow r_i \in 2^{D_i^1 \times \cdots \times D_i^{n_i}})\}.$$

The information content of an M-table consists of two components: a collection of a set of definite mixed relations, among which at least one is the real world truth, correlating to the minimal models of the underlying first-order theory represented by the sure component of the M-table and a set of mixed maybe tuples.

Before presenting the formal definition of the information content of M-tables, the following notation is necessary: a mixed tuple set u is said to be a (proper) subset of another mixed tuple set v if every member of u is a (proper) subset of the corresponding member of v.

Definition 5.1.1: Let $MR=<R_1,R_2,\cdots,R_k>$ be an M-relational scheme and T be an M-table over MR, then

$\Gamma_{MR} = \{T \mid T$: $M-table\ over\ MR\}$, and

$\Sigma_{MR} = \{<U,v> \mid U$: $set\ of\ M-relations\ over\ MR, v$: $M-relation\ over\ MR\}$.

The information content of an M-table T over MR, where $T_{sure}=\{<u_{11},\ldots,u_{1k}>,\ldots,<u_{n1},\ldots,u_{nk}>\}$, is a mapping REP from Γ_{MR} to Σ_{MR} such that: $REP(T) = REDUCEREP(<MM,M>(T))$, where

(1) $<MM,M>$ is a mapping from Γ_{MR} to Σ_{MR} such that: $<MM,M>(T) = <MM(T),M(T)>$,

$MM(T) = \{<r_1,\cdots,r_k> \mid (\exists d_1)(\exists x_1)\cdots(\exists d_n)(\exists x_n)$

$((\forall i)(1\le i\le n \rightarrow (1\le d_i\le k \wedge x_i\in u_{id_i})) \wedge$

$<r_1,\cdots,r_k>=collate^k(<x_1,\cdots,x_n>,<d_1,\cdots,d_n>))\}$, and $M(T) = T_{maybe}$.

where $collate^k(<x_1,\cdots,x_n>,<d_1,\cdots,d_n>)$ is a function that returns a multi-relation $<r_1,\cdots,r_k>$ by placing x_i in r_{d_i}, $1\le i\le n$.

(2) $REDUCEREP$ is a mapping from Σ_{MR} to Σ_{MR}. Suppose that $MR=<R_1,\cdots,R_k>$ and $<U,v>\in\Sigma_{MR}$ such that $v=<v_1,\cdots,v_k>$, then

$REDUCEREP(<U,v>)=<U^o,v^o>$, where

$U^o = \{<r_1,\cdots,r_k> \mid <r_1,\cdots,r_k>\in U \wedge \neg(\exists s_1)\ldots(\exists s_k)$

$(<s_1,\cdots,s_k>\in U \wedge <s_1,\cdots,s_k>\subset<r_1,\cdots r_k>)\}$, and

$v^o = <v_1^o,\cdots,v_k^o>$, where

$v_j^o=\{t \mid (t\in v_j \vee (\exists s_1^1)\cdots(\exists s_k^1)(\exists s_1^2)\ldots(\exists s_k^2)$

$(<s_1^1,\ldots,s_k^1>\in U \wedge <s_1^2,\ldots,s_k^2>\in U \wedge$

$<s_1^1,\cdots,s_k^1>\subset<s_1^2,\cdots,s_k^2> \wedge t\in(s_j^2-s_j^1))) \wedge$

$\neg(\exists r_1)\cdots(\exists r_k)(<r_1,\cdots,r_k>\in U^o \wedge t\in r_j)\}, 1\le j\le k.$

$MM(T)$ corresponds to the (nonminimal) models of the underlying first-order theory of the M-table T and $M(T)$ corresponds to the maybe tuples of T. On the other hand, U^o corresponds

to the minimal models of the underlying first-order theory. An example of the mapping $<MM,M>$ and *REDUCEREP* is illustrated in Figure 5.2.

5.2 Redundancies in M-tables

For notational convenience, the union (difference) of two mixed tuple sets u and v denotes the union (difference) of each member of u with its corresponding member of v. Suppose that $T=<T_{sure},T_{maybe}>$ is an M-table defined over the scheme $MR=<R_1,\cdots,R_k>$, the following two kinds of redundancies could exist across the different components of T:

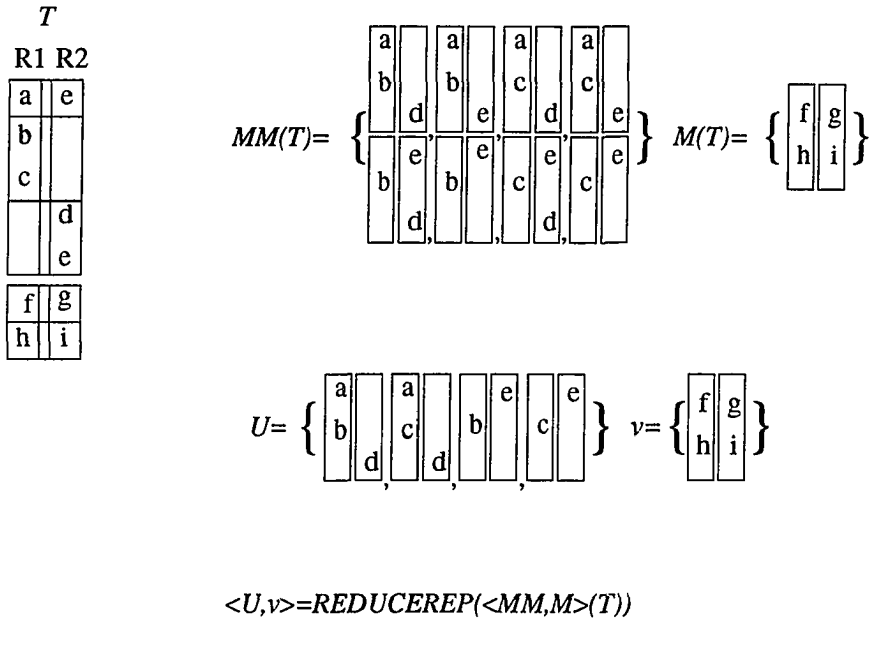


Figure 5.2: *REDUCEREP* and $<MM,M>(T)$

1. A mixed tuple set of T_{sure} subsumes another mixed tuple set of T_{sure}. That is, there exists $u=<u_1,\cdots,u_k>\in T_{sure}$ and $v=<v_1,\cdots,v_k>\in T_{sure}$ such that $u\subset v$. This redundancy is removed by deleting v from T_{sure} and including $(v-u)$ in T_{maybe}.

2. A mixed tuple set of T_{sure} contains a mixed tuple of T_{maybe}. That is, there exists $t, t \in r_i$ *of* $<r_1, \cdots, r_k>$ *of* T_{maybe} and $u = <u_1, \cdots, u_k> \in T_{sure}$ such that $t \in u_i$, for some $i, 1 \leq i \leq k$. This redundancy is removed by simply removing t from r_i.

For instance, the mixed tuple $<a,b>$ of the mixed relation T in Figure 5.3 is redundant because of the presense of $<a,\emptyset>$. Furthermore, $<c,d>$ of T_{sure} contains the tuple c corresponding to R_1 of T_{maybe}.

The above mentioned redundancies can be removed by the operator called *REDUCE* as defined below:

Definition 5.2.1 Let T be an M-table over the scheme $MR = <R_1, \cdots, R_k>$, where $T_{maybe} = <r_1, \cdots, r_k>$. Then $REDUCE(T) = T'$, where

$$T'_{sure} = \{u \mid u \in T_{sure} \wedge \neg(\exists v)(v \in T'_{sure} \text{ and } v \subset u)\},$$

$$T'_{maybe} = <v'_1, \cdots, v'_k>, \text{ and}$$

$$v'_j = \{t \mid (t \in r_j \vee (\exists u)(\exists s)(u = <u_1, \cdots, u_k> \in T_{sure} \wedge$$

$$s = <s_1, \cdots, s_k> \in T_{sure} \wedge u \subset s \wedge t \in (s_j - u_j))) \wedge$$

$$\neg(\exists w)(w = <w_1, \cdots, w_k> \in T_{sure} \wedge t \in w_j)\}, \ 1 \leq j \leq k.$$

The *REDUCE* operator is also illustrated in Figure 5.3.

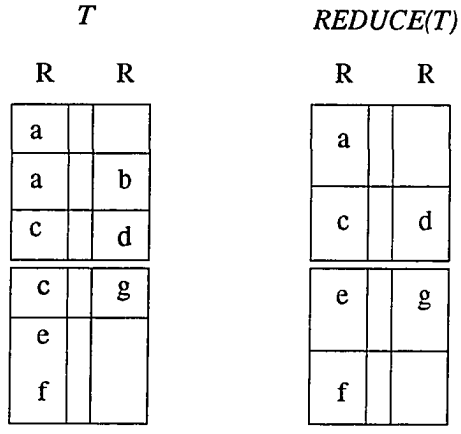


Figure 5.3: Redundancies and *REDUCE*

5.3 Updates on M-tables and Their Semantics

5.3.1 The notion of correctness

As has been defined earlier, the mapping *REP* maps an M-table, T, over scheme MR, to elements of Σ_{MR}. $REP(T)$ is essentially an incomplete database, which contains a set of instances of mixed relations, and at least one of the instances is the real world truth. On the other hand, M-tables are representations of incomplete databases, and it needs to be assured that the representation is able to represent the results of any meaningful operations (e.g., join, insertion). Consider an operator f. In order to extend f to operate on M-tables, it must be assured that the extended operator captures the effect of the corresponding operator on the various definite mixed relations represented in the information content of M-tables. This notion of correctness can still be expressed by Figure 3.2 and is formally defined as follows.

Definition 5.3.1.1 Let f be an extended operator on M-tables, and T be an arbitrary M-table. Suppose that f_Σ is defined on Σ_R and f_Γ is defined on Γ_R. Then f_Γ is semantically correct if:

(1) $REP(f_\Gamma(T)) = f_\Sigma(REP(T))$, for unary f_Γ, and

(2) $REP(f_\Gamma(T_1, T_2)) = f_\Sigma(REP(T_1), REP(T_2))$, for binary f_Γ.

5.3.2 Extended update operations on M-tables

Suppose for the discussion follows that T is an M-table over M-relation scheme $MR = <R_1, \cdots, R_k>$ and t_i as well as t_{ij} are tuples over R_i, $1 \le i \le k$. Since M-tables encompass horizontal and vertical indefinite information as well as maybe information, insertion operations on M-tables must be able to incorporate both such knowledge into the corresponding database. Furthermore, the sure component of an M-table could contain a collection of mixed tuple sets, one should also be able to insert a new mixed tuple into a mixed tuple set of the sure component. The consolidation of new horizontal indefinite information corresponds to the insertion of a mixed tuple and tuple set into the sure component of an M-table, while the incorporation of new

vertical indefinite information is signified by the insertion of a mixed tuple set. Hence, there are four types of insertions on M-tables:

(1) $\mathbf{INS}^{\Gamma}_{sure}[<t_1, t_2, \cdots, t_k>](T)$, where there exists at least one $i, 1 \le i \le n$, such that $t_i \neq \varnothing$ - inserting a mixed tuple into the sure component.

(2) $\mathbf{INS}^{\Gamma}_{sure}[\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T)$, where there exits at least one $i, 1 \le i \le k$ such that $t_{ij} \neq \varnothing, 1 \le j \le n$ - inserting a mixed tuple set into the sure component.

(3) $\mathbf{INS}^{\Gamma}_{sure}[<t_1, \cdots, t_k>, \{<t_{11}, \cdots, t_{1k}>, \cdots, <t_{n1}, \cdots, t_{nk}>\}](T)$, where $n \ge 2$, and there exists at least one $i, 1 \le j \le n$, such that $t_i \neq \varnothing$ and $\{<t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\} \in T_{sure}$ - inserting a mixed tuple into a mixed tuple set of the sure component.

(4) $\mathbf{INS}^{\Gamma}_{maybe}[<t_1, \cdots, t_k>](T)$, where there exists at least one $i, 1 \le i \le n$, such that $t_i \neq \varnothing$ - inserting a maybe tuple into the maybe component.

The deletion operations on M-tables can be extended symmetrically to the extended insertion operations, and the notation is as follows:

(1) $\mathbf{DEL}^{\Gamma}_{sure}[<t_1, \cdots, t_k>](T)$, where $\{<t_1, \ldots, t_k>\} \in T_{sure}$ - deleting a mixed tuple from the sure component.

(2) $\mathbf{DEL}^{\Gamma}_{sure}[\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T)$, where $\{<t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\} \in T_{sure}$ - deleting a mixed tuple set from the sure component.

(3) $\mathbf{DEL}^{\Gamma}_{sure}[<t_{i1}, \ldots, t_{ik}>, \{<t_{11}, \ldots, t_{1k}>, \ldots, <t_{i1,\ldots,t_{ik}}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}](T)$, where $n \ge 3$, and $\{<t_{11}, \ldots, t_{1k}>, \ldots, <t_{i1}, \ldots, t_{ik}>, \ldots, <t_{n1}, \ldots, t_{nk}>\} \in T_{sure}$ - deleting a mixed tuple from a mixed tuple set of the sure component.

(4) $\mathbf{DEL}^{\Gamma}_{maybe}[<t_1, \cdots, t_k>](T)$, where $\{<t_1, \ldots, t_k>\} \in T_{maybe}$ - deleting a mixed tuple from the maybe component.

The extended modifications, which can be expressed in terms of a sequence of deletions and

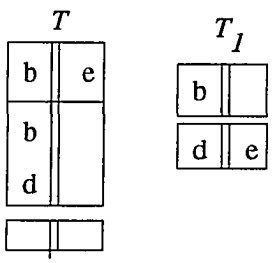
insertions, will not be formalized.

5.3.3 Semantics of the extended update operations

The focus of this section is that of analyzing the underlying semantics of the extended update operations on M-tables. Let T be an M-table and $<U,v>=REP(T)$ be the information content of the M-table T. U, the underlying incomplete database corresponding to T, then is the set of all possible mixed relations represented by T and of which at least one is the real world truth, yet it is not known which one or ones are true. On the other hand, v is the set of all the mixed maybe tuples of T. According to this underlying semantics, or information content, of M-tables, any operator f on M-tables should be interpreted as if f is applied to all the possible state of the underlying incomplete database. This interpretation is used as the foundation for establishing the semantic correctness of the extended update operations on M-tables.

First let us consider inserting a mixed tuple $<t_1,...,t_k>$ into the sure component of T. Based on the underlying semantics, each t_i, $1\leq i\leq k$, should be inserted separately into its corresponding relation of all the possible states of U to guarantee that it is in the true state or states of the real world. The example in Figure 5.4 brings out this semantics.

Attempting to extract the underlying semantics for the insertion of a mixed tuple set $W=\{<t_{11},...,t_{1k}>,...,<t_{n1},\ldots,t_{nk}>\}$, the following observation and analogy may be made. The tuple set W to be inserted may be viewed as a first-order theory and corresponding to W is a collection of mixed relations W_m which is the model associated with W (e.g., the information content of W). Therefore, it is equivalent to insert a set of mixed relations W_m into U. In terms of logic, this type of insertion can be viewed as a logic "and" of two disjunctions. One of the disjunction is U and the other is W_m. In light of the distributivity of $\wedge$ over $\vee$, this implies the insertion of each mixed relation into each of its corresponding possible state of U. An example is shown in Figure 5.4.



$$T_1 = \mathbf{INS}_{sure}[<b,\varnothing>](T)$$

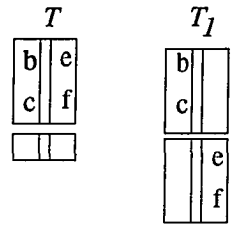
$$U = \{ \boxed{b}\boxed{\ } , \boxed{d}\boxed{e} \} \qquad v = \varnothing$$

$$<U,v> = REP(T)$$

$$U_1 = \{ \boxed{b}\boxed{\ } \} \qquad v_1 = \{ \boxed{d}\boxed{e} \}$$

$$<U_1,v_1> = REP(T_1)$$

Figure 5.4: An Example of the Extended Insertion Operation



$$T_1 = \mathbf{INS}_{sure}[<d,\varnothing>,\{<b,\varnothing>,<d,\varnothing>\}](T)$$

$$U = \{\boxed{b}\boxed{\ }, \boxed{c}\boxed{\ }, \boxed{\ }\boxed{e}, \boxed{\ }\boxed{f}\} \qquad v = \varnothing$$

$$<U,v> = REP(T)$$

$$U_1 = \{\boxed{b}\boxed{\ }, \boxed{c}\boxed{\ }\} \qquad v_1 = \{\boxed{e}\boxed{f}\}$$

$$<U_1,v_1> = REP(T_1)$$

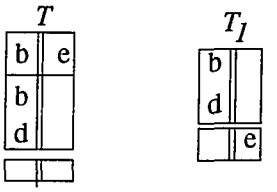
Figure 5.5: An Example of the Extended Insertion Operation

Compounding the analysis is the insertion of a mixed tuple $<t_1, \cdots t_k>$ into a mixed tuple set $\{<t_{11}, \cdots t_{1k}>, \cdots, <t_{n1}, \cdots, t_{nk}>\}$ of an M-table T. This type of insertion amplifies the degree of uncertainty. This is true because the combination of the possibility that $<t_1,...,t_k>$ being true true has to be taken into account on top of all the other possibilities of valid combinations of $\{<t_{11},...,t_{1k}>,...,<t_{n1},...,t_{nk}>\}$ are true. The semantics of this type of insertions is best understood when placed in perspective relative to that of the insertion and deletion of a mixed tuple set. That is, it is equivalent to inserting the mixed tuple set:

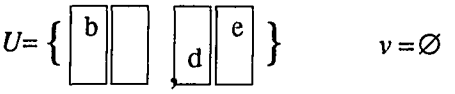
$\{<t_1, \cdots t_k>,...,<t_{11}, \cdots t_{1k}>, \cdots ,<t_{n1}, \cdots ,t_{nk}>\}$ into

$T - \{<t_{11}, \ldots, t_{1k}>,...,<t_{n1}, \ldots, t_{nk}>\}$.

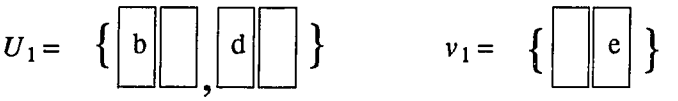
For example, inserting the mixed tuple $<d,\varnothing>$ into the mixed tuple set $<b,e>$ as shown in Figure 5.7 is equivalent to first deleting $<b,e>$ from T and then inserting $\{<d,\varnothing>,<b,e>\}$. An example is illustrated in Figure 5.6.



$$T_1 = \mathbf{INS}_{sure}[<d,\varnothing>,\{<b,\varnothing>,>d,\varnothing>\}](T)$$



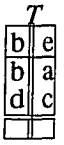
$$<U,v> = REP(T)$$

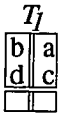


$$<U_1,v_1> = REP(T_1)$$

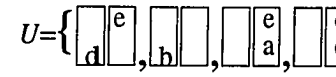
Figure 5.6: An Example of the Extended Insertion Operation

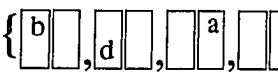
The other component of $REP(T)$, v, is a single relation which contains the tuples that may be true in the real world. Thus the insertion of a mixed tuple $<t_1, \cdots ,t_k>$ into the maybe component of T, are similar to insertions under conventional relations and should be interpreted as inserting $t_i, 1 \le i \le k$ into its corresponding counterpart in v.



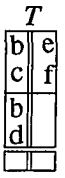


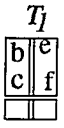
$T_1 = \mathbf{DEL}_{sure}[<b,e>](T)$

$U=\{$  $\}$ $v=\varnothing$

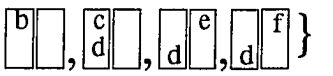
$U_1=\{$  $\}$ $v_1=\varnothing$

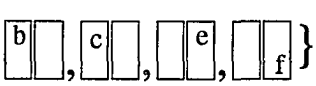
$<U,v>=REP(T)$ $<U_1,v_1>=REP(T_1)$



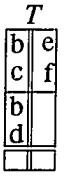


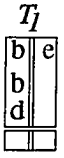
$T_1 = \mathbf{DEL}_{sure}[\{<b,\varnothing>,<d,\varnothing>\}](T)$

$U=\{$  $\}$ $v=\varnothing$

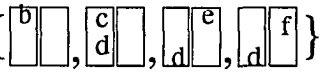
$U_1=\{$  $\}$ $v_1=\varnothing$

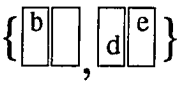
$<U,v>=REP(T)$ $<U_1,v_1>=REP(T_1)$





$T_1 = \mathbf{DEL}_{sure}[<b,e>,\{<b,e>,<c,f>\}](T)$

$U=\{$  $\}$ $v=\varnothing$

$U_1=\{$  $\}$ $v_1=\varnothing$

$<U,v>=REP(T)$ $<U_1,v_1>=REP(T_1)$

Figure 5.7: An Example of the Extended Deletion Operation

The semantics of the deletion operations mirror that of the insertion operations and the detailed analysis is omitted. However, examples are shown in Figure 5.7. Based on the above discussion, the semantics of the extended insertion and deletion operations are formally defined as follows.

Definition 5.3.3.1: Let T be an M-table over $<R_1, \ldots, R_k>$, then

(1) $\mathbf{INS}^{\Sigma}_{sure}[<t_1, \cdots, t_k>](<MM(T), M(T)>) = REDUCEREP < U', v' >$, where

$$U' = MM(T) \cup MM(<t_1, \cdots, t_k>) = \{ <r_1, \cdots, r_k> | (\exists u_1)\ldots(\exists u_k)(\exists i)((<u_1,\ldots,u_k> \in MM(T)) \wedge (1 \le i \le k) \wedge (r_1 = u_1) \wedge \cdots \wedge (r_i = u_i \cup t_i) \wedge \cdots \wedge (r_k = u_k)) \}$$

and $v' = M(T)$.

(2) $\mathbf{INS}^{\Sigma}_{sure}[\{<t_{11}, \cdots, t_{1k}>,\ldots,<t_{n1}, \cdots, t_{nk}>\}]$

$(<MM(T), M(T)>) = REDUCEREP < U', v' >$, where

$$U' = MM(T) \cup MM(\{<t_{11},\ldots,t_{1k}>,\ldots,<t_{n1},\ldots,t_{nk}>\}) = \{ <r_1, \cdots, r_k> | (\exists u_1)\ldots(\exists u_k)(\exists w_1)\ldots(\exists w_k)((<u_1,\ldots,u_k> \in MM(T)) \wedge (<w_1,\ldots,w_k> \in MM(\{<t_{11}, \ldots, t_{1k}>,\ldots,<t_{n1},\ldots,t_{nk}>\}) \wedge (r_1 = u_1 \cup w_1) \wedge \cdots \wedge (r_k = u_k \cup w_k)) \}$$ and $v' = M(T)$.

(3) $\mathbf{INS}^{\Sigma}_{maybe}[<t_1, \cdots, t_k>](<MM(T), M(T)>) =$

$REDUCEREP < U', v' >$, where $U' = MM(T)$, and

$$v' = M(T) \cup M(<t_1, \cdots, t_k>) = \{ <v_1 \cup t_1, \cdots, r_k \cup t_k> \}$$

(4) $\mathbf{DEL}^{\Sigma}_{sure}[<t_1, \cdots, t_k>](<MM(T), M(T)>) = REDUCEREP < U', v' >$, where

$$U' = MM(T) - MM(<t_1, \cdots, t_k>) = \{ <r_1, \cdots, r_k> | (\exists u_1)\ldots(\exists u_k)((<u_1,\ldots,u_k> \in MM(T)) \wedge (r_1 = u_1 - t_1) \wedge \cdots \wedge (r_k = u_k - t_k)) \}$$ and $v' = M(T)$.

(5) $\mathbf{DEL}^{\Sigma}_{sure}[\{<t_{11}, \cdots, t_{1k}>,\ldots,<t_{n1}, \cdots, t_{nk}>\}]$

$(<MM(T), M(T)>) = REDUCEREP < U', v' >$, where

$$U' = MM(T) - MM(\{<t_{11},\ldots,t_{1k}>,\ldots,<t_{n1},\ldots,t_{nk}>\}) =$$

$$\{<r_1,\cdots,r_k> \mid (\exists u_1)\ldots(\exists u_k)(\exists w_1)\ldots(\exists w_k)((<u_1,\ldots,u_k> \in MM(T)) \wedge$$

$$(<w_1,\ldots,w_k> \in MM(\{<t_{11},\ldots,t_{1k}>,\ldots,<t_{n1},\ldots,t_{nk}>\}) \wedge$$

$$(r_1 = u_1 - w_1) \wedge \cdots \wedge (r_k = u_k - w_k))\} \text{ and } v' = M(T).$$

(6) $\mathbf{DEL}^{\Sigma}_{maybe}[<t_1,\cdots,t_k>](<MM(T),M(T)>) =$

$$REDUCEREP<U',v'>, \text{ where } U' = MM(T), \text{ and}$$

$$M(T)' = v - M(<t_1,\cdots,t_k>) = \{<v_1 - t_1,\cdots,r_k - t_k>\}$$

(7) $\mathbf{INS}^{\Sigma}_{sure}[<t_1,\cdots,t_k>,\{<t_{11},\cdots,t_{1k}>,\ldots,<t_{n1},\cdots,t_{nk}>\}](<MM(T),M(T)>)) =$

$$\mathbf{INS}^{\Sigma}_{sure}[U_2](\mathbf{DEL}^{\Sigma}_{sure}[U_1](<U,v>)), \text{ where}$$

$U_1 = \{<t_{11},\cdots,t_{1k}>,\cdots,<t_{n1},\cdots,t_{nk}>\}$, and

$U_2 = \{<t_1,\cdots,t_k>,<t_{11},\cdots,t_{1k}>,\cdots,<t_{n1},\cdots,t_{nk}>\}$

(8) $\mathbf{DEL}^{\Sigma}_{sure}[<t_{i1},\cdots,t_{ik}>,\{<t_{11},\cdots,t_{1k}>,\ldots,<t_{i1},\ldots,t_{ik}>,\ldots,<t_{n1},\cdots,t_{nk}>\}]$

$$(<MM(T),M(T)>))$$

$= \mathbf{INS}^{\Sigma}_{sure}[U_2](\mathbf{DEL}^{\Sigma}_{sure}[U_1](<U,v>))$, where

$U_1 = \{<t_{11},\cdots,t_{1k}>,\cdots,<t_{i1},\ldots,t_{ik}>,\ldots,<t_{n1},\cdots,t_{nk}>\}$,

$U_2 = \{<t_{11},\cdots,t_{1k}>,\cdots,<t_{(i-1)1},\ldots,t_{(i-1)k}>,$

$<t_{(i+1)1},\ldots,t_{(i+1)k}>,\ldots,<t_{n1},\cdots,t_{nk}>\}$.

5.3.4 Extending update operations on M-tables

In this section, the syntactic definitions for the extended update operations on M-tables are presented. The syntactic definitions are devised based on the semantics discussed earlier and the equivalence of the syntactic and semantic definitions, that is the correctness of the extended update operations, will be established in the next section. Examples illustrating the syntactic definitions are embedded in Figure 5.4 - Figure 5.7.

Definition 5.3.4.1 : Let T be an M-table over M-relational scheme $<R_1,\ldots,R_k>$. Then:

(1) $\mathbf{INS}^{\Gamma}_{sure}[<t_1, t_2, \cdots, t_k>](T) = REDUCE(T')$, where $T'_{sure} = T_{sure} \cup \{t_1, \cdots, t_k\}$, and $T'_{maybe} = T_{maybe}$

(2) $\mathbf{INS}^{\Gamma}_{sure}[\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T)$ $= REDUCE(T')$, where $T'_{sure} = T_{sure} \cup \{<t_{11}, \cdots, t_{1k}>, \cdots, <t_{n1}, \cdots, t_{nk}>\}$, and $T'_{maybe} = T_{maybe}$

(3) $\mathbf{INS}^{\Gamma}_{sure}[<t_1, \cdots, t_k>, \{<t_{11}, \cdots, t_{1k}>, \cdots, <t_{n1}, \cdots, t_{nk}>\}](T) = REDUCE(T')$, where $T'_{sure} = (T_{sure} - \{<t_{11}, \ldots, t_{1k}>, \cdots, <t_{n1}, \cdots, t_{nk}>\}) \cup \{<t_1, \cdots, t_k>, <t_{11}, \ldots, t_{1k}>, \cdots <t_{n1}, \cdots, t_{nk}>\}$, and $T'_{maybe} = T_{maybe}$

(4) $\mathbf{INS}^{\Gamma}_{maybe}[<t_1, \cdots, t_k>](T) = REDUCE(T')$, where $T'_{sure} = T_{sure}$, and $T'_{maybe} = T_{maybe} \cup \{<t_1, \cdots, t_k>\}$.

Definition 5.3.4.2 : Let T be an M-table over M-relational scheme $<R_1, \ldots, R_k>$. Then:

(1) $\mathbf{DEL}^{\Gamma}_{sure}[<t_1, \cdots, t_k>](T) = REDUCE(T')$, where $T'_{sure} = T_{sure} - \{t_1, \cdots, t_k\}$, and $T'_{maybe} = T_{maybe}$

(2) $\mathbf{DEL}^{\Gamma}_{sure}[\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T) =$ $REDUCE(T')$, where $T'_{sure} = T_{sure} - \{<t_{11}, \cdots, t_{1k}>, \cdots, <t_{n1}, \cdots, t_{nk}>\}$, and $T'_{maybe} = T_{maybe}$

(3) $\mathbf{DEL}^{\Gamma}_{sure}[<t_{i1}, \cdots, t_{ik}>, \{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{i1}, \cdots, t_{ik}>, \cdots, <t_{n1}, \cdots, t_{nk}>\}](T) = REDUCE(T')$, where $T'_{sure} = (T_{sure} - \{<t_{11}, \ldots, t_{1k}>, \cdots, <t_{i1}, \ldots, t_{ik}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}) \cup \{<t_{11}, \ldots, t_{1k}>, \ldots, <t_{(i-1)1}, \ldots, t_{(i-1)k}>, <t_{(i+1)1}, \ldots, t_{(i+1)k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}$, and $T'_{maybe} = T_{maybe}$

(4) $\mathbf{DEL}^{\Gamma}_{maybe}[<t_1, \cdots, t_k>](T) = REDUCE(T')$, where $T'_{sure} = T_{sure}$, and $T'_{maybe} = T_{maybe} - \{<t_1, \cdots, t_k>\}$.

5.4 Results and Properties of the Update Operations on M-tables

Two claims are made in this section. Firstly, the semantic correctness of the extended update operations of insertion and deletion is claimed. Secondly, some properties associated with the extended insertion and deletion operators are determined.

Claim 5.4.1: Let T be a reduced M-table over M-relational scheme $<R_1, \ldots, R_k>$, and $<U,v>=REDUCEREP(<MM(T),M(T)>)$. Then

(1) $\mathbf{INS}^{\Sigma}_{sure}[<t_1, \ldots, t_k>](<MM(T),M(T))=<MM,M>(\mathbf{INS}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](T))$,

(2) $\mathbf{INS}^{\Sigma}_{sure}[\{<t_{11}, \cdots, t_{1k}>,\ldots,<t_{n1}, \ldots, t_{nk}>\}]$

$(<MM(T),M(T)>)=<MM,M>$

$(\mathbf{INS}^{\Gamma}_{sure}[\{<t_{11}, \ldots, t_{1k}>, \cdots, <t_{n1}, \ldots, t_{nk}>\}](T))$,

(3) $\mathbf{INS}^{\Sigma}_{sure}[<t_1, \ldots, t_k>),REP(\{<t_{11}, \ldots, t_{1k}>,\ldots,<t_{n1}, \ldots, t_{nk}>\}]$

$(<MM(T),M(T)>)=<MM,M>$

$(\mathbf{INS}^{\Gamma}_{sure}[<t_1, \ldots, t_k>,\{<t_{11}, \ldots, t_{1k}>, \cdots, <t_{n1}, \ldots, t_{nk}>\}](T))$,

(4) $\mathbf{INS}^{\Sigma}_{maybe}[<t_1, \ldots, t_k>](<MM(T),M(T)>)=$

$<MM,M>(\mathbf{INS}^{\Gamma}_{maybe}[<t_1, \ldots, t_k>](T))$,

(5) $\mathbf{DEL}^{\Sigma}_{sure}[<t_1, \ldots, t_k>](<MM(T),M(T)>)=$

$<MM,M>(\mathbf{DEL}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](T))$,

(6) $\mathbf{DEL}^{\Sigma}_{sure}[\{<t_{11}, \ldots, t_{1k}>, \cdots, <t_{n1}, \ldots, t_{nk}>\}](<MM(T),M(T)>)$

$<MM,M>(\mathbf{DEL}^{\Gamma}_{sure}[\{<t_{11},t_{1k}>, \cdots, <t_{n1},\ldots,t_{nk}>\}](T))$,

(7) $\mathbf{DEL}^{\Sigma}_{sure}[<t_{i1},\ldots,t_{ik}>,\{<t_{11}, \ldots, t_{1k}>,\ldots,$

$<t_{i1}, \ldots, t_{ik}>, \cdots, <t_{n1}, \ldots, t_{nk}>\}](<MM(T),M(T)>)=$

$<MM,M>(\mathbf{DEL}^{\Gamma}_{sure}[<t_{i1}, \ldots, t_{ik}>,\{<t_{11},\ldots,t_{1k}>,\ldots,$

$<t_{i1}, \ldots, t_{ik}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T))$,

(8) $\mathbf{DEL}^{\Sigma}_{maybe}[<t_1, \ldots, t_k>](<MM(T),M(T)>)=$

$<MM,M>(\mathbf{DEL}^{\Gamma}_{maybe}[<t_1, \ldots, t_k>](T))$

Claim 5.4.1 is illustrated in Figure 5.5 - Figure 5.8.

For conventional relational databases, inserting a tuple into a relation and then deleting it yields the original relation. This property also holds if deletion is performed first and followed by insertion. However, the update operations of insertion and deletion on M-tables exhibit different behaviors as established in the following claim.

Claim 5.4.2: Let T be a reduced M-table over M-relational scheme $<R_1, \ldots, R_k>$, and $<U,v>=REDUCEREP(<MM(T),M(T)>)=REP(T)$. Then

(1) $\mathbf{DEL}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](\mathbf{INS}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](T)) \neq T$

(2) $\mathbf{INS}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](\mathbf{DEL}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](T)) = T$

(3) $\mathbf{DEL}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](\mathbf{INS}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](T)) \neq$

$\mathbf{INS}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](\mathbf{DEL}^{\Gamma}_{sure}[<t_1, \ldots, t_k>](T))$

(4) $\mathbf{DEL}^{\Gamma}_{sure}[<\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}]$

$(\mathbf{INS}^{\Gamma}_{sure}[\{t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T)) \neq T$

(5) $\mathbf{INS}^{\Gamma}_{sure}[<\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}]$

$(\mathbf{DEL}^{\Gamma}_{sure}[\{t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T)) = T$

(6) $\mathbf{DEL}^{\Gamma}_{sure}[<\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}]$

$(\mathbf{INS}^{\Gamma}_{sure}[\{t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T)) \neq$

$\mathbf{INS}^{\Gamma}_{sure}[<\{<t_{11}, \cdots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}]$

$(\mathbf{DEL}^{\Gamma}_{sure}[\{t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \cdots, t_{nk}>\}](T))$

(7) $\mathbf{DEL}^{\Gamma}_{sure}[t,u'](\mathbf{INS}^{\Gamma}_{sure}[t,u](T)) \neq T$, where

$t=<t_1, \ldots, t_k>$, $u=\{<t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}$, and

$u'=\{<t_1, \ldots, t_k>, <t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}$

(8) $\mathbf{INS}^{\Gamma}_{sure}[t,u'](\mathbf{DEL}^{\Gamma}_{sure}[t,u](T)) = T$, where

$t=<t_1, \ldots, t_k>$,

$u=\{<t_1, \ldots, t_k>, <t_{11}, \ldots, t_{1k}>, \ldots, <t_{n1}, \ldots, t_{nk}>\}$, and

$$u' = \{ < t_{11}, \ldots, t_{1k} >, \ldots, < t_{n1}, \ldots, t_{nk} > \}$$

(9) $\mathbf{DEL}^{\Gamma}_{sure}[t,u'](\mathbf{INS}^{\Gamma}_{sure}[t,u](T)) \neq \mathbf{INS}^{\Gamma}_{sure}[t,u](\mathbf{DEL}^{\Gamma}_{sure}[t,u'](T)) = T$, where

$$t = < t_1, \ldots, t_k >, u = \{ < t_{11}, \ldots, t_{1k} >, \ldots, < t_{n1}, \ldots, t_{nk} > \}, \text{ and}$$

$$u' = \{ < t_1, \ldots, t_k >, < t_{11}, \ldots, t_{1k} >, \ldots, < t_{n1}, \ldots, t_{nk} > \}$$

(10) $\mathbf{DEL}^{\Gamma}_{maybe}[< t_1, \ldots, t_k >](\mathbf{INS}^{\Gamma}_{maybe} < t > (T)) =$

$\mathbf{INS}^{\Gamma}_{maybe}[< t_1, \ldots, t_k >](\mathbf{DEL}^{\Gamma}_{maybe}[< t_1, \ldots, t_k >](T)) = T$

6. MODELING EXCLUSIVELY DISJUNCTIVE INFORMATION

One of the problems associated with various null value approaches proposed in the literature, as pointed out in Chapter 2, is that they only allow a limited form of incomplete information to be represented. A more serious/fatal deficiency of some of the proposed models for null values lies in the fact that they do not support the relational operators of selection, projection, cartisian product, union, difference, and join in a semantically correct manner.

One direction for the further research therefore consists of examining the possibility of generalizing the null value approach to represent the most general forms of incomplete information. The limitation of the expressive power of the existing null values approaches is resulted from their underlying treatment of incomplete information on the attribute level. Therefore, a more general model should handle incomplete information on the tuple level.

The focus of the research described in this chapter is that of extending the relational model to represent exclusively disjunctive information. That is, disjunctions of the form $P_1 \mid P_2 \mid \cdots \mid P_n$, where $\mid$ denotes a generalized logical exclusive *or* indicating that exactly one of P_i's can be true. Note that the generalized exclusive *or* represents an *if* $\cdots$ *then* $\cdots$ conjunction in the sense that if P_i is true and $P_1 \mid P_2 \mid \cdots \mid P_i \mid \cdots \mid P_n$ is true, then all the P_j's other than P_i are false, $1 \le i \le n$, $1 \le j \le n$ and $j \ne i$. Specifically, an extended relational model called *E-table* for capturing exclusively disjunctive information is introduced and preliminary results with respect to the E-table model are presented.

With the introduction of E-tables, the formalization of the semantics of relational operators then becomes the most important issue. A major portion of this chapter is dedicated to this issue. Specifically, the relational operators of selection, projection, cartisian product, difference, union, and intersection are extended to E-tables. All of these operators are extended in a semantically correct manner and are faithful in the sense that they reduce to the usual relational

operators when the E-tables consist of only complete information.

To pave the way for such an extension of E-tables, however, the following issues need to be resolved: (1) how to correctly handle negative information, (2) how to identify and remove redundancies, and (3) how to identify and resolve contradictions?

There is a fundamental relationship between negation and exclusive indefiniteness as will be discussed later in this chapter. In a Horn database, Reiter proposed the notion of the Closed World Assumption and has shown that negative information can be assumed to be true straightforwardly if its positive counterpart cannot be proven [Reit78]. However, in non-Horn databases, the notion of the Generalized Closed World Assumption suggested by Minker and the concept of minimal models must be used [Mink82]. In this chapter, the notion of the Generalized Closed World Assumption for non-Horn databases of E-tables (GCWAE) is defined. GCWAE is a natural evolution and generalization of GCWA [Mink82] since the generalized exclusive *or* can be viewed as a logical counterpart of the inclusive *or* (evolution) and each disjunct in the exclusive disjunction of E-tables can itself be a conjunction (generalization). Redundancies within E-tables in turn are closely related to negative information and must be resolved under the context of GCWAE.

In the remainder of this chapter, we first present in Section 6.1 some background knowledge regarding E-tables and then formally define the semantics of E-tables. Section 6.2 covers the treatment of negative information in the context of E-table structure and discusses the notion of GCWAE. Section 6.3 is devoted to the resolution of redundancies in E-tables and the issues associated with contradictions are also analyzed. Finally, Section 6.4 formally defines the extended relational operators of selection, projection, cartisian product, difference, union, and intersection on E-tables in a semantically correct manner.

6.1 E-tables and Exclusively Disjunctive Information

The idea behind E-tables is to use sets as compound tuple values to represent exclusively incomplete information. Here a set is composed of sets of tuple sets and carries three dimensions of semantics. The first dimension offers the meaning that the actual value is *exactly one* of the elements (tuple sets) of the set (i.e., this particular tuple set is true). The second dimension presents the interpretation that every tuple of a tuple set is an actual value if its corresponding tuple set is true (i.e., every tuple of a true tuple set is true). Finally, the third dimension is expressed by special dummy values (denoted by the set $\lambda=\{\varepsilon_1,\varepsilon_2,...,\}$) indicating that there may be no actual value (i.e., nothing is true).

Informally, an *E-table* structure T over a relational scheme $R=<A_1,A_2,\cdots,A_k>$ consists of two components, namely the *definite component* T_D and the (exclusively) *indefinite component* T_E. The definite component is a set of (conjunctive) tuples over R, each of which is known to be true. The indefinite component consists of (conjunctive) sets of tuple sets such that each set of tuple sets is known to be true. However, it is not known which tuple set is true within each set of tuple sets. Note that the exclusive indefiniteness implies that one and only one tuple set within a set of tuple sets can be true.

The indefinite component models exclusive disjunction of the form $P_1 \mid P_2 \mid \cdots \mid P_n$, where P_i, $1\le i\le n$, is either an atomic formula or a conjunction of atomic formulas and $\mid$ denotes a generalized n-ary exclusive *or* operator such that $P_1 \mid P_2 \mid \cdots \mid P_n$ is true if *exactly one* of the P_i's, $1\le i\le n$, is true. The generalization is generalization is necessary since the regular logical exclusive or $\bar{\vee}$ is defined based on the number of true/false values. For example, $1 \bar{\vee} 1 \bar{\vee} 0=0$ while $1 \bar{\vee} 1 \bar{\vee} 1=1$. Table 6.1 shows the truth table of the generalized ternary exclusive *or* $\mid$.

Table 6.1: Generalized Ternary Exclusive *or*

a	*b*	*c*	*a*	*b*	*c*
0	0	0		0	
0	0	1		1	
0	1	0		1	
0	1	1		u	
1	0	0		1	
1	0	1		u	
1	1	0		u	
1	1	1		u	

Table 6.2: Extended $\wedge$, $\vee$, and $\neg$

	$\neg$
0	1
1	0
u	u

$\wedge$	0	1	u
0	0	0	u
1	0	1	u
u	u	u	u

$\vee$	0	1	u
0	0	1	u
1	1	1	u
u	u	u	u

The truth value of $P_1 \mid P_2 \mid \cdots \mid P_n$ is undefined (denoted by "u" as shown in Table 6.1) if more than one of the P_i's are true, $1 \leq i \leq n$. The usual logical operations of $\wedge$, $\vee$, and $\neg$ are extended with respect to "u" as shown in Table 6.2.

Consider, for example, the E-table **PART** of Figure 6.1. The definite information that parts $P1$ and $P2$ are of *black* and *white* respectively can be represented by the definite component as $\{(P1, black), (P2, white)\}$. The exclusive indefinite information that part $P3$ is of either *red* or *blue* and either $P4$ or $P5$ is of *green* can be represented by the indefinite component as $\{\{(P3, red), (P3, blue)\}, \{(P4, green), (P5, green)\}\}$. The **SUPPLIER** E-table represents the fact that supplier $S1$ supplies part $P1$, $S2$ supplies either $P2$ or supplier $S2$ supplies both $P3$ and $P4$, and either $P3$ supplies $P5$ is true or nothing is true.

PART

p1 black p2 white
p3 red p3 blue
p4 green p5 green

SUPPLIER

s1 p1
s2 p2 s2 p3, s2 p4
s3 p5 ε

Figure 6.1: Motivating Examples

Under the context of logic, E-tables can be viewed as a conjunction of ground atomic formulas. For example, the definite and indefinite components of E-table **PART** represents the following information (ground formulas):

(1) **PART**$(P1, black)$

(2) **PART**$(P2, white)$

(3) **PART**$(P3, red) \mid$ **PART**$(P3, blue)$

(4) **PART**($P4$,*green*) | **PART**($P4$,*green*).

Formally, an E-table T over a relational scheme $R=<A_1,A_2,\ldots,A_n>$ with associated domains D_i, $1\leq i\leq n$, is defined as $T = <T_D,T_E>$, where

$$T_D \subseteq (D_1\times D_2\times\cdots\times D_n\cup\lambda),$$

$$T_E \subseteq 2^{2^{(D_1\times\cdots\times D_n)\cup\lambda}} -$$

$$\left(\{\varnothing\}\cup\{\{\varnothing\}\}\cup\{\{s\}|s\in(2^{(D_1\times\cdots\times D_n)\cup\lambda})\}\cup\{\{\varnothing,s\}|s\in(2^{(D_1\times\cdots\times D_n)\cup\lambda})\}\right)$$

where $\lambda=\{\varepsilon_1,\varepsilon_2,\varepsilon_3,.....\}$ is an infinite collection of marked special dummy values which are assumed distinct unless they have the same subscript. Dummy values are used to indicate that there may be no actual values.

Before presenting the formal definition of the information content of E-tables, the following notation is necessary: let R be a relational scheme, then

$\Gamma_R = \{T|\, T$: *E−table over* $R\}$, and

$\Sigma_R = \{U|\, U$:set of relations over $R\}$.

Following Imielinski and Lipski's notation [ImLi84], the information content of E-tables is denoted by *REP*. *REP* is a mapping from Γ_R to Σ_R and consists of a collection of all the possible definite relations, of which *exactly one* is the real world truth, represented by the definite and indefinite component of an E-table. The formal definition of *REP* is first offered and the essence of the definition will then be explained and discussed through examples shown in Figure 6.2

Definition 6.1.1 Let T be a consistent (i.e., does not contain any contradictions) E-table over relational scheme R such that $T=<T_D,T_E>$, where $T_E = \{W_1,\ldots,W_n\}$.

$$REP(T)=\{T_D\cup W|\, W=w_1\cup w_2\cup\cdots\cup w_n\ \wedge$$

$$(\forall i)(1\leq i\leq n\rightarrow w_i\in W_i)\wedge\neg\left[(\exists u)(u\in\bigcup_{i=1}^{n}(W_i-w_i)\wedge(u\subseteq W)\right]\}$$

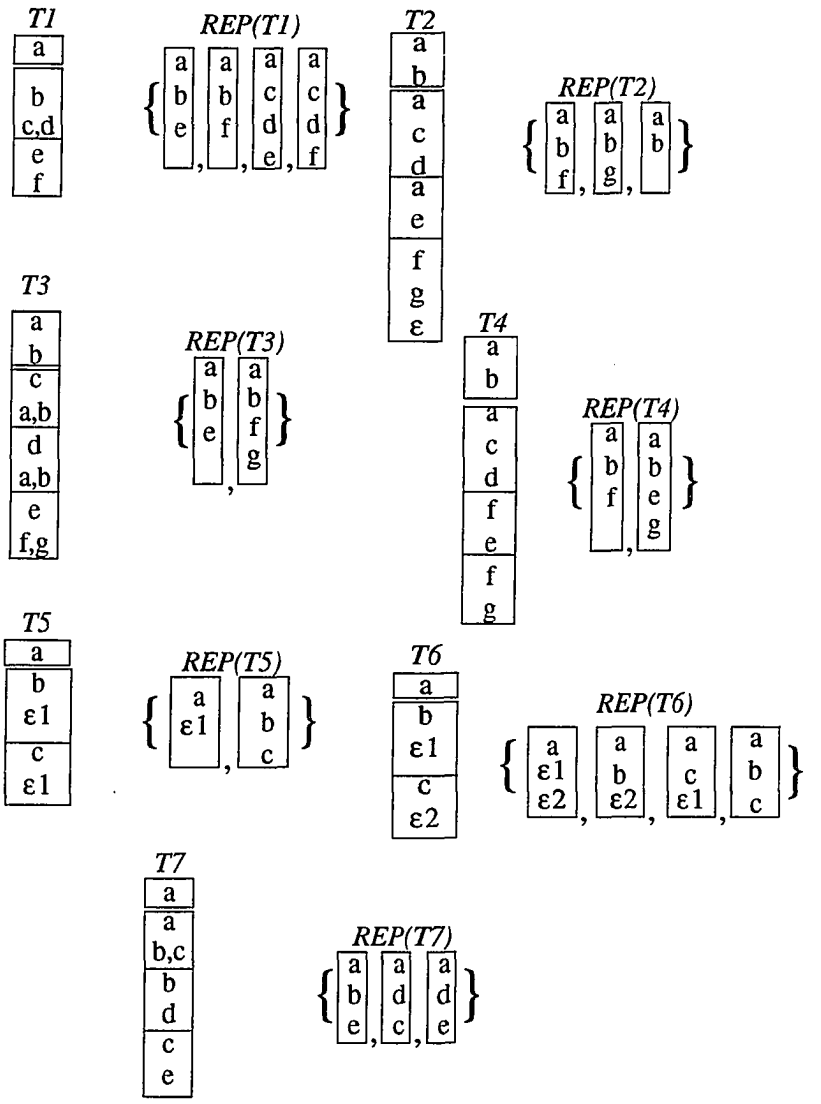


Figure 6.2: *REP*(*T*) - The Information Content of *T*

Examples of the mapping *REP* are given in Figure 6.2. The information content of the E-table *T*1 is straightforward and is just the enumeration of all the possible combination of the real world truth. For *T*2, only *a* can be chosen from the first and second sets of tuple sets of the indefinite component since *a* is in the definite component and is always true. Similarly, this restriction is true for the first and second sets of tuple sets in the indefinite component of *T*3. In

$T4$, if f is chosen from the second set of tuple sets of the indefinite component, then f must be chosen from the third set of tuple sets. $T5$ has the similar restriction as the one described for $T4$ above except that the common element is a special dummy value. Comparing to $T5$, $T6$ has more possible real world truth than $T5$ although it has almost the same structure as $T5$. This is due to the fact that the restriction that applies to $T5$ does not apply to $T6$. Lastly, the E-table $T7$ exemplifies that the fact that a is always true implies that a must be chosen from the first set of tuple sets in the indefinite component. Therefore, b and c cannot be chosen at the same time.

Note that the relations of the information content of E-tables could contain special dummy values (e.g., $REP(T5)$ and $REP(T6)$) which are used as placeholders. The purpose of including dummy values in the information content of E-tables is to simplify the definitions for relational operators such as selections and unions. In reality, dummy values are insignificant when they are co-existent with real domain values. Specifically, for any given E-table T over relational scheme R and any relation $r=\{t_1,t_2,\ldots,t_n\}$ of $REP(T)$, if $t_i\in\lambda$, $1\le i\le n$, and t_j's are real domain tuple values, $1\le j\le n$ and $j\neq i$, then r is equivalent to $\{t_1,t_2,\ldots,t_{i-1},t_{i+1},\ldots,t_n\}$. On the other hand, if all the t_i's are dummy values, $1\le i\le n$, then r is equivalent to the empty set $\varnothing$.

Therefore, the following definition is necessary:

Definition 6.1.2 Let T be a consistent E-table over relational scheme R and $REP(T)$ be the information content of T, then

$$REPREP(REP(T))=\{r|(\exists r_1)(r_1\in REP(T)\wedge r=\{t|(t\in r_1\wedge t\notin\lambda)\})\}.$$

Given $REP(T)$, $REPREP(REP(T))$ represents all the different real world possibilities, with dummy values properly displaced, modeled by the E-table T. Among all the possibilities in $REPREP(REP(T))$, only one of them is the real world truth. Examples of $REPREP$, for E-tables $T5$ and $T6$ of Figure 6.2, are pictured in Figure 6.3.

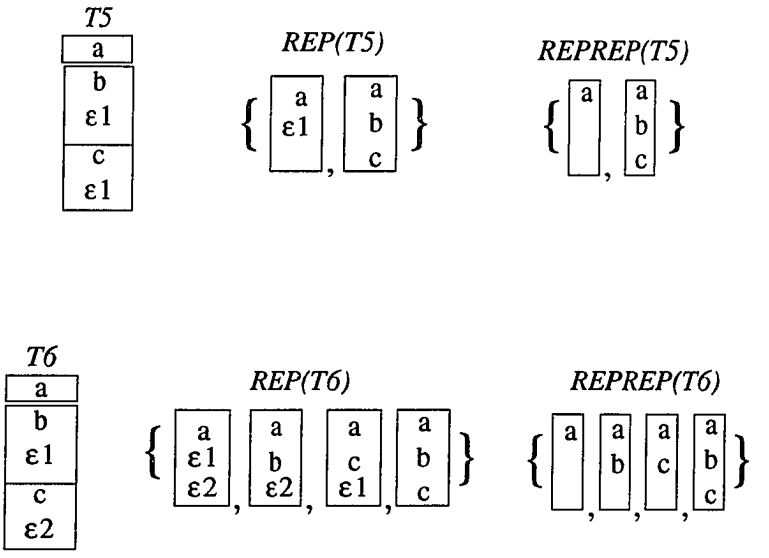


Figure 6.3: *REPREP*(*T*)

6.2 The Closed World Assumption and E-tables

This section concerns with the treatment of negative information under the E-table setting. In particular, the notion of the Generalized Closed World Assumption for non-Horn databases of E-tables (GCWAE) is defined. A database is Horn, or definite, if every ground atomic formula is Horn while a ground atomic formula is Horn if it has at most one positive atom when written in its corresponding disjunctive form. On the contrary, a database is non-Horn, or indefinite, if it contains non-Horn ground atomic formulas, which have more than one positive atoms when written in its corresponding disjunctive form. GCWAE is a natural evolution and generalization of GCWA (Minker [Mink82]) for the reasons that the generalized exclusive *or* can be viewed as a logical counterpart of the inclusive *or* and each disjunct in the exclusively disjunction can itself be a conjunction. GCWAE also incorporates the extended GCWA of Yahya and Henschen [YaHe85] and allows indefinite negative information. For the discussion that follows a database

is viewed as a conjunction of ground atomic formulas. Specifically, A database of E-tables is viewed as a conjunction and each conjunct corresponds to an E-tables of the database and is in turn a conjunction representing the definite and indefinite components of the E-table.

In retrospect, the Closed World Assumption (CWA) concept was introduced by Reiter for Horn databases in [Reit78]. Under CWA, positive facts, or atomic formulas, are explicitly stored in the database and negative facts are implicitly present provided that their corresponding positive counterpart cannot be proven from the positive facts explicitly stored in the database. Semantically, a negative ground formula is true under CWA if its counterpart is not in the minimal model of the database (a minimal model for Horn database is the intersection of all its models). It has been shown that Horn databases are consistent with CWA. The objective of CWA is to avoid storing potentially overwhelming amount of negative information explicitly in the database.

However, CWA introduces inconsistencies under non-Horn databases. For instance, let $DB = \{P_a \vee P_b\}$, then both $\overline{P}_a$ and $\overline{P}_b$ are true since they cannot be derived from DB. Consequently, $DB \cup \{\overline{P}_a, \overline{P}_b\}$ is inconsistent. Minker in [Mink82] proposed an extension of the CWA, the Generalized Closed World Assumption (GCWA) for non-Horn databases based on the concept of minimal models. A minimal model for a non-Horn database is a model of that database such that no proper subset of that model is also a model. Semantically, GCWA defines that a negative fact can be assumed to be true if its positive counterpart is not in any minimal models of the underlying first-order theory of its corresponding database. GCWA is consistent with non-Horn databases and is faithful in the sense that it reduces to CWA for Horn databases.

GCWA was described by Minker relative to non-Horn databases consisting of inclusive indefinite information. Considering the simple database similar to the one as used by Minker in [Mink82]:

$$DB = \{P_a \vee P_b, P_c \vee \overline{P}_d, P_e\}$$

The following are the models of the above database:

$$\{\{P_a,P_e\},\{P_b,P_e\},\{P_a,P_b,P_e\},\{P_a,P_c,P_e\},\{P_b,P_c,P_e\},$$
$$\{P_a,P_b,P_c,P_e\},\{P_a,P_c,P_d,P_e\},\{P_b,P_c,P_d,P_e\},\{P_a,P_b,P_c,P_d,P_e\}\}$$

The following are then the minimal models of the above database:

$$\{\{P_a,P_e\},\{P_b,P_e\}\}$$

Therefore, $\overline{P}_c$ and $\overline{P}_d$ are true because their counterparts are not in any of the minimal models (the set of such negative facts is denoted by $\overline{\overline{DB}}$ for a non-Horn database DB. That is, $\overline{\overline{DB}}=\{\overline{P}_c,\overline{P}_d\}$). The extended GCWA of Yahya and Henschen [YaHe85] admits indefinite negative information and states that a ground negative clause $C=\overline{P}_1 \vee \overline{P}_2 \vee \cdots \vee \overline{P}_n$ can be assumed to be true if C is true in every minimal models of the underlying first-order theory. For instance, the ground negative clauses $\overline{P}_c \vee \overline{P}_d$, $\overline{P}_a \vee \overline{P}_b$, and $\overline{P}_b \vee \overline{P}_c \vee \overline{P}_e$ would be included in $\overline{\overline{DB}}$ under the extended GCWA. Note that assuming $\overline{P}_a \vee \overline{P}_b$ to be true is similar to interpreting the logical *or* in $P_a \vee P_b$ to be exclusive rather than inclusive.

The foregoing discussion has summerized the ideas leading to the definition of GCWAE: Let T be an E-table over the relational scheme R, then a ground negative conjunction $C=\neg(P(t_1) \wedge P(t_2) \wedge \cdots \wedge P(t_n))$ can be assumed to be true if C is true in every minimal models of the underlying first-order theory corresponding to T, where P is the predicate corresponding to T, and t_i's, $1 \le i \le n$, are tuples over R. GCWAE for a database of E-tables is then the union of the negative information (e.g., the set of C's) of each E-table of the database.

By way of example, consider the following database of E-tables:

$$DB=\{(P_a \mid (P_b \wedge \overline{P}_c)),P_d\}$$

The following are the models of the above database:

$$\{\{P_a,P_d\},\{P_b,P_d\},\{P_a,P_c,P_d\},\{P_a,P_b,P_c,P_d\}\}\}$$

The following are then the minimal models of the above database:

$$\{\{P_a,P_d\},\{P_b,P_d\}\}$$

Therefore:

$$\overline{\overline{DB}} = \{\{\neg P_c\},\{\neg(P_a \wedge P_b)\},\{\neg(P_a \wedge P_c)\},\{\neg(P_b \wedge P_c)\},$$
$$\{\neg(P_d \wedge P_c)\},\{\neg(P_a \wedge P_b \wedge P_c)\},\{\neg(P_a \wedge P_b \wedge P_d)\},$$
$$\{\neg(P_b \wedge P_c \wedge P_d)\},\{\neg(P_a \wedge P_b \wedge P_c \wedge P_d)\}\}$$

Notice that DB and GCWAE is consistent, which is also illustrated by the following example.

$$DB = \{P_a \mid (\varepsilon_1 \wedge P_c) \mid (\overline{P}_d \wedge P_e)\}$$

The following are the models of the above database:

$$\{\{P_a,P_e\},\{\varepsilon_1,P_e\},\{P_a,P_c,P_d,P_e\},\{\varepsilon_1,P_c,P_d,P_e\}\}$$

The minimal models are as follows:

$$\{\{P_a,P_e\},\{\varepsilon_1,P_e\}\}$$

Therefore, $\overline{\overline{DB}} = \{\overline{P}_c,\overline{P}_d\}$ and $DB \cup \overline{\overline{DB}}$ is consistent.

The significance of GCWAE is that it assures that the negative information associated with a given E-table can be implicitly represented.

6.3 E-tables and Redundancies

To use E-tables in a practical and meaningful fashion, the following problems must be solved: what constitute contradictions and redundancies in E-tables and how to resolve them? Redundancies and contradictions could arise with the addition of new information as a result of, for example, the union and the insertion operations. Suppose that T is a database table over R which is free of any redundancies and contradictions. A new tuple t of R to be incorporated into T in general falls into one of the four possible categories:

(1) t has already been represented in T and therefore t should be ignored.

(2) t is not represented by the existing information in T and therefore should be added to T.

(3) t overlaps with the existing information in T and therefore should replace the existing information in T.

(4) t contradicts with the existing information in T.

Case 3 and 4 are considered to be redundancies and contradictions respectively. The precise characterization of what constitutes contradictions in E-tables will not be explored in detail here. However, several possible types of contradictions that could occur in an E-table are identified below:

(1) At least two tuple sets of a set of tuple set W of T_E are subsets of T_D. That is, there exists a $W \in T_E$ and there exists $w_1 \in W$ and $w_2 \in W$ such that $w_1 \subseteq T_D$ and $w_2 \in T_D$. For example, $(a \wedge b) \wedge (a \mid b)$ is a contradiction since $a \mid b$ implies that a and b cannot be both true at the same time.

(2) Recursive statements as source of contradictions. For example, $(a \mid b) \wedge (b \mid c) \wedge (c \mid a)$ is a contradiction since if a is true in $(a \mid b)$, then $(b \mid c)$ implies that only c can be true. However, c is true in $(c \mid a)$ implies a must not be true.

Assume for the discussion follows that the E-tables are consistent (free of contradictions). For simplicity, also assume that there is an operator *CONTRADICTION* for both the elements of Σ_R and Γ_R. *CONTRADICTION*(T) returns T itself if an E-table T is consistent and $\varnothing$ otherwise. Similarly, *CONTRADICTION*(U) returns U of Σ_R if it is consistent and $\varnothing$ otherwise.

The exercise of extracting the conditions under which redundancies arise in E-tables and consequently removing redundancies from E-tables involves two areas. The first one is concerned with identifying and removing redundancies between the definite and the indefinite component, and redundancies existing within the indefinite component. The second area is concerned with removing redundancies associated with special dummy values. For the discussion follows, the following definition is necessary:

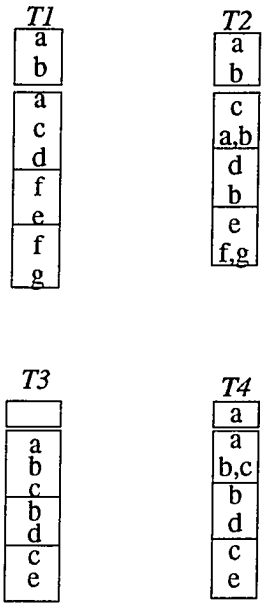


Figure 6.4: Entanglement

Definition 6.3.1.: Let T be an E-table with $T_E = \{W_1, W_2, ..., W_n\}$, then a tuple set w of W_i, $1 \leq i \leq n$, *entangles* T_E if there exists at least one $t \in w$, there exists a $w \in W_j$, $1 \leq j \leq n$ and $j \neq i$, such that $t \in w$. The W_i and all the W_j's are called the *entanglement* of the indefinite component (T_E).

Figure 6.4 can be used to illustrate this concept: the tuple set $\{f\}$ in the sets of tuple sets $\{\{f\},\{e\}\}$ and $\{\{f\},\{g\}\}$ entangles the indefinite component of the E-table $T1$, while the set of tuple sets $\{\{c\},\{a,b\}\}$ and $\{\{d\},\{b\}\}$ forms the entanglement for the E-table $T2$ (because of the tuple set $\{a,b\}$ and $\{b\}$). For the E-table $T3$, the set of tuple set $\{\{a\},\{b\},\{c\}\}$ entangles the set of tuple set $\{\{b\},\{d\}\}$ (because of the tuple set $\{b\}$) and $\{\{c\},\{e\}\}$ (because of the tuple set $\{c\}$) respectively. Finally, in the E-table $T4$, the set of tuple sets $\{\{a\},\{b,c\}\}$ entangles the indefinite component since the other two sets of tuple sets have b and c as their members.

With respect to **consistent** E-tables, the following nine kinds of redundant information could occur:

(1) A tuple set w of an indefinite set of tuple sets W of T_E is a proper subset of the definite component (e.g., $T1$, $T2$, and $T3$ of Figure 6.5).

(2) An indefinite set of tuple sets W_1 of T_E is a subset of another indefinite set of tuple sets W_2 of T_E (e.g., $T3$ of Figure 6.5).

(3) A tuple t of T_D appears in a non-singleton tuple set w_i of a set of tuple sets W of T_E. That is, $t \in w_i$ and $w_i \in W = \{w_1, \cdots w_n\}$, where $n \geq 1$, and $|w_i| > 1$ (e.g., $T4$ of Figure 6.5)..

(4) A tuple t is a member of all the tuple sets of a set of tuple set (e.g., $T5$ of Figure 6.5).

(5) A tuple set w_1 of a set of tuple is a subset of another tuple set w_2 in the same set of tuple set. That is, there exists w_1 and w_2 with $w_1 \in W$ and $w_2 \in W$, where $W \in T_E$, such that $w_1 \subset w_2$ (e.g., $T4$ of Figure 6.5).

(6) A set of tuple sets W of T_E contains more than one singleton tuple set containing a special dummy value as its member and at least two of the dummy values are not a part of any entanglement (e.g., $T6$ of Figure 6.5).

(7) A set of tuple set W of T_E consists entirely of singleton tuple set containing a special dummy value and at least two of the dummy values are not a part of any entanglement (e.g., $T7$ of Figure 6.5).

(8) A tuple set in a set of tuple set contains dummy values which are not a part of any entanglement and the non-dummy values of this tuple set is not a subset of any other tuple sets in the same set of tuple sets (e.g., ε_1 and ε_2 of $T8$ of Figure 6.5).

(9) The indefinite component contains entanglements.

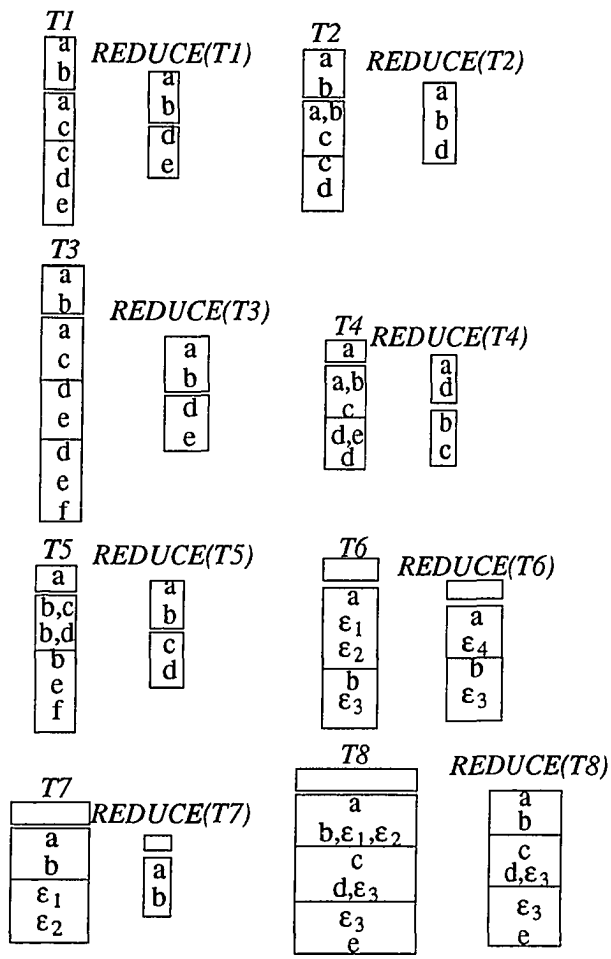


Figure 6.5: *REDUCE*

To illustrate the resolution of redundancies, consider the first case through an example. Let T be an E-table with $T_D = \{a\}$ and $T_E = \{\{a\},\{b,c\}\}$, and let the domain of T be $\{a,b,c\}$. In this case, the tuple set $\{a\}$ is a subset of T_D. In terms of logic, the definite and indefinite component of T represents the conjunction $a \wedge (a \mid (b \wedge c))$. Intuitively, since a is true and thus $(b \wedge c)$ cannot be true. Therefore, T can be reduced to the E-table T' with $T'_D = \{a\}$ and $T'_E = \varnothing$. However, intuitions may be deceiving sometimes as the foregoing claim can easily be

invalidated by Table 6.3.

Table 6.3 brings out the fact that the inequivalence occurs when $(b \wedge c)$ is true. However, under GCWAE, $b \wedge c$ is assumed to be always false with respect to $T' = <\{a\}, \varnothing>$. This amounts to saying that if $b \wedge c$ is false, then the above two tables are equivalent. The above discussion prompts for the following definition.

Table 6.3: $a \wedge (a \mid (b \wedge c)) \neq a$

a	b	c	$a \mid (b \wedge c)$	$a \wedge (a \mid (b \wedge c))$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	0
1	0	0	1	1
1	0	1	1	1
1	1	0	1	1
1	1	1	u	u

Definition 6.3.2 Let T_1 and T_2 be E-tables over scheme R, then $T_1 =_E T_2$ if $T_1 = T_2$ except where the truth value is "u".

Table 6.3 can also be used to illustrate that $a =_E a \wedge (a \mid (b \wedge c))$ since they are equivalent when $b \wedge c$ is false (eighth line in Table 6.2). Another way of perceiving this is that $REP(\{\{a\}, \{\{a\}, \{b,c\}\} = REP(\{\{a\}, \varnothing\}$. The following two theorems formally establish this equivalence.

Theorem 6.3.1 $a \wedge (b \mid c) =_E (a \wedge b) \mid (a \wedge c)$.

For the discussions follow, the following theorem is necessary.

Theorem 6.3.2 The following equivalence holds:

$$a_1 \mid a_2 \mid \cdots \mid a_n =_E (a_1 \mid a_2 \mid \cdots \mid a_{n-1}) \mid a_n$$

Theorem 6.3.3 The following equivalence holds:

$a \wedge (a \mid B_1 \mid \ldots\ldots \mid B_n) =_E a,$

where a is an atomic ground formula representing a singleton tuple and B_i, $1 \leq i \leq n$, is a conjunction of ground formulas which are free of a's.

In general, if a tuple set w of some set of tuple sets W of the indefinite component of an E-table is a subset of the definite component of that E-table and the tuple sets of $W-w$ do not entangle the indefinite component, then W should be removed from the indefinite component. If the tuple sets of $W-w$ entangle the indefinite component, then in addition to removing W from the indefinite component, every tuple of $W-w$ should be removed from their corresponding sets of tuple sets as well.

Table 6.4: $(a \mid b) \wedge (a \mid b \mid c) =_E (a \mid b)$

a	b	c	$a \mid b$	$a \mid b \mid c$	$(a \mid b) \wedge (a \mid b \mid c$
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	1	1
0	1	1	1	u	u
1	0	0	1	1	1
1	0	1	1	u	u
1	1	0	u	u	u
1	1	1	u	u	u

The second type of redundancies can be resolved similarly. For example, $(a \mid b) \wedge (a \mid b \mid c) =_E (a \mid b)$ as vindicated by Table 6.4. This is a generalization of Theorem 6.3.2 as shown by Theorem 6.3.3.

Theorem 6.3.4 The following equivalence holds:

$A \wedge (A \mid B_1 \mid \ldots\ldots \mid B_n) =_E A$

where A is a conjunction of atomic ground formulas and each conjunct can be a generalized exclusive disjunction (e.g., $A = a \wedge (b \mid c)$), and each B_i, $1 \leq i \leq n$, is a conjunction of ground

formulas which does not contain the elements of A.

Similarly, if a set of tuple set W_1 is a subset of another set of tuple sets W_2 and the tuple sets of $W-w$ do not entangle the indefinite component, then W_2 should be removed from the indefinite component. If the tuple sets of W_2-W_1 entangle the indefinite component, then in addition to removing W_2 from the indefinite component, every tuple of W_2-W_1 should be removed from their corresponding sets of tuple sets.

The third type of redundancies is formalized by Theorem 6.3.5.

Theorem 6.3.5 The following equivalence holds:

$$a \wedge ((a \wedge b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m) =_E$$
$$a \wedge ((b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m)$$

where a and b_i's, $1 \le i \le n$, are atomic ground formulas and Each C_i, $1 \le i \le n$, is a conjunction of ground formulas which are free of a and b_i's, $1 \le i \le n$.

Similarly, the fourth type of redundancies can be formalized by the following theorem.

Theorem 6.3.6 The following equivalence holds:

$$(a \wedge b_1) \mid (a \wedge b_2) \mid \ldots\ldots \mid (a \wedge b_n) =_E a \wedge (b_1 \mid \ldots\ldots \mid b_n)$$

where each b_i, $1 \le i \le n$, is a conjunction of ground formulas.

In general, the third and fourth types of redundancies can be resolved by moving the tuple t from the tuple set w_i to the definite component.

With respect to the fifth type of redundancies, the following theorem can be established.

Theorem 6.3.7 The following equivalence holds:

$$(a_1 \wedge \ldots\ldots \wedge a_n) \mid (a_1 \wedge \ldots\ldots \wedge a_n \wedge B) =_E (a_1 \wedge \ldots\ldots \wedge a_n)$$

where B is a conjunction of ground atomic formulas which is free of a_i's, $1 \le i \le n$.

In general, if a tuple set w_1 of some set of tuple sets W of the indefinite component of an E-table is a subset set of another tuple set w_2 of the same set of tuple set W and the tuple sets of w_2-w_1 do not entangle the indefinite component, then w_2 should be removed from the

indefinite

component. If the tuple sets of $w_2 - w_1$ entangle the indefinite component, then in addition to removing w_2 from the indefinite component, every tuple of $w_2 - w_1$ should be removed from their corresponding sets of tuple set as well.

The sixth and seventh types of redundancies should be resolved by removing all but one special dummy values that do not appear in any other sets of tuple sets, while the eighth type of redundancies can be resolved by removing the dummy values from the tuple set it resides provided that it is not the only value in that tuple set.

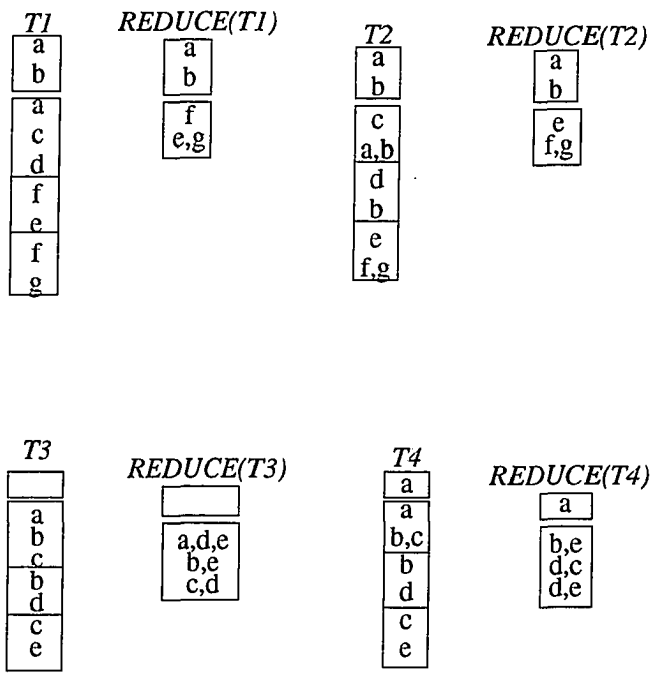


Figure 6.6: E-table Restructuring

The last type of redundancies involves entanglement and should be resolved by rearranging the indefinite component. Specifically, let T be an E-table and $T_E=\{W_1,\ldots,W_n\}$. Suppose that $W_{k_1},W_{k_2},\ldots,W_{k_l}$, $1\le k_i\le n$, where $1\le i\le l$ and $1\le l\le n$, constitute an entanglement, then T is to be restructured as follows:

1. All the sets of tuple set of T_E, except $W_{k_1},W_{k_2},\ldots,W_{k_l}$ remain unchanged, and
2. $W_{k_1},W_{k_2},\ldots,W_{k_l}$ are to be merged into one set of tuple set W', which is obtained as follows:

$$W'=\{r\mid(\exists r_1)(r_1\in REP(T)\wedge r=r_1\cap(W_{k_1}\cup W_{k_2}\cup\cdots\cup W_{k_l})\wedge r\neq\varnothing)\}.$$

Figure 6.6 illustrates the restructuring of E-tables due to entanglements.

Alternatively, the resolution of entanglement can be defined by precisely specifying each tuple set. Suppose that T ia an E-table and $T_E=\{W_1,\ldots,W_n\}$, where $W_i=\{w_{i1},\ldots,w_{ik_i}\}$, $1\le i\le n$. Consider the simplest case of entanglement: there exists $W_i=\{w_{i1},\ldots,w_{il},\ldots,w_{ik_i}\}$ and $W_j=\{w_{j1},\ldots,w_{jm},\ldots,w_{jk_j}\}$ such that $w_{il}=w_{jm}$, where $1\le i\le n$ and $1\le j\le n$. Then, T_E is restructured as follows:

1. All the sets of tuple set of T_E, except W_i and W_j, remain unchanged, and

2. W_i and W_j are to be merged into one set of tuple set W' as follows:

$$\begin{aligned}w'=\{&\{w_{i1},w_{j1}\},\ldots,\{w_{i1},w_{j(m-1)}\},\{w_{i1},w_{j(m+1)}\},\ldots,\{w_{i1},w_{jk_j}\},\ldots\\&\{w_{i(l-1)},w_{j1}\},\ldots,\{w_{i(l-1)},w_{j(m-1)}\},\{w_{i(l-1)},w_{j(m+1)}\},\ldots,\{w_{i(l-1)},w_{jk_j}\},\ldots\\&\{w_{i(l+1)},w_{j1}\},\ldots,\{w_{i(l+1)},w_{j(m-1)}\},\{w_{i(l+1)},w_{j(m+1)}\},\ldots,\{w_{i(l+1)},w_{jk_j}\},\ldots\\&\{w_{ik_i},w_{j1}\},\ldots,\{w_{ik_i},w_{j(m-1)}\},\{w_{ik_i},w_{j(m+1)}\},\ldots,\{w_{ik_i},w_{jk_j}\},\ldots\{w_{il}\}\}\end{aligned}$$

Note that the alternative definition is quite complicated. It becomes more and more tedious and difficult as the degree of entanglements gets more and more complicated. Further, the alternative definition is sensitive to whether an E-table contains redundancies or not. In other

word, if an E-table contains redundancies, then the result obtained by first resolving redundancies and then resolving entanglement is different from the result yielded by first resolving entanglement and then resolving redundancies. This phenomenon is illustrated in Figure 6.7. The first definition, however, does not have this drawback. In addition, it is simpler. Therefore, the first definition will be used from now on.

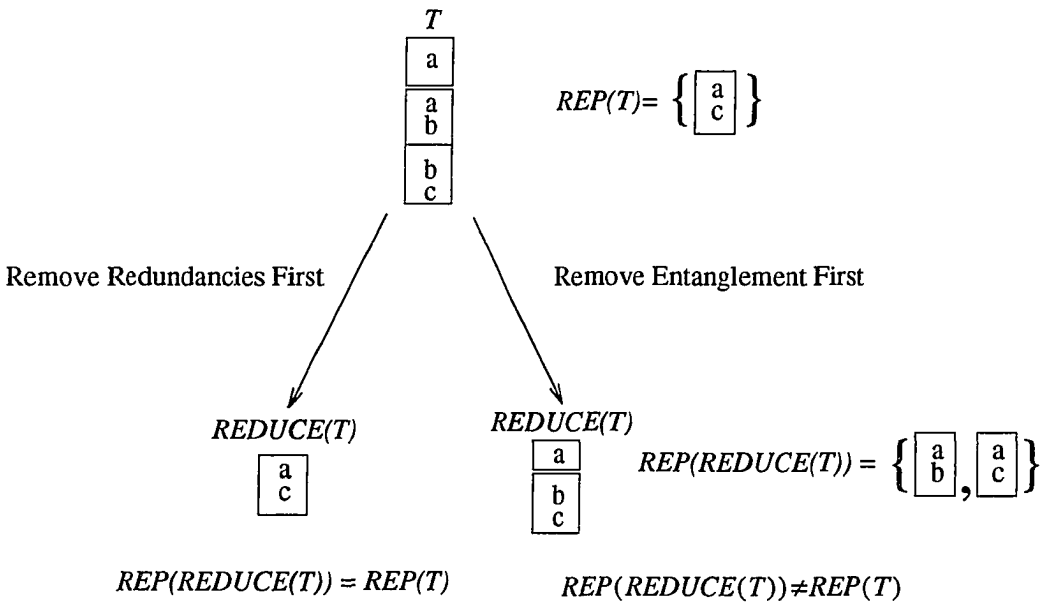


Figure 6.7: the Order Sensitiveness of the Alternative Definition

Let *REDUCE* be an operator which removes the redundancies from E-tables. *REDUCE*(T) applies the applicable reduction steps described above to an E-table T until no more reductions are possible. Figure 6.5 also illustrates some examples of *REDUCE*.

The following theorem states that *REDUCE* is idempotent.

Theorem 6.3.8: For any consistent E-table $T \in \Gamma_R$, $REDUCE(REDUCE(T)) = REDUCE(T)$.

The following theorem establishes the relationship between *REP* and $=_E$.

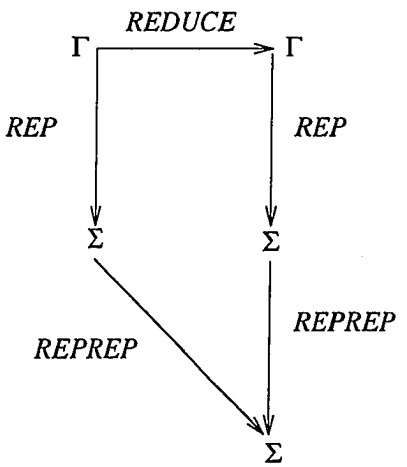
Theorem 6.3.9: For any consistent E-tables $T_1 \in \Gamma_R$ and $T_2 \in \Gamma_R$, if $T_1 =_E T_2$, then $REP(T_1) = REP(T_2)$

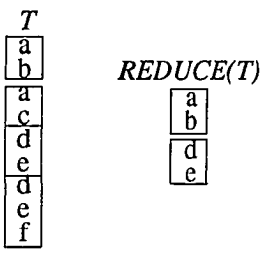
The following theorem validates that *REDUCE* neither creates nor destroys any information.

Theorem 6.3.10: For any consistent E-table $T \in \Gamma_R$,

$$REPREP(REP(REDUCE(T))) = REPREP(REP(T)).$$

Theorem 6.3.10 is illustrated in Figure 6.8.





$$REPREP(REP(REDUCE(T))) = REPREP(REP(T)) = \left\{ \begin{array}{|c|}\hline a \\ \hline b \\ \hline d \\ \hline \end{array}, \begin{array}{|c|}\hline a \\ \hline b \\ \hline e \\ \hline \end{array} \right\}$$

Figure 6.8: $REP(REDUCE(T) = REP(T)$

6.4 Relational Algebra and E-tables

In this section, the relational operators of projection, selection, cartisian product, difference, union, and intersection are extended to E-tables. For each relational operator, both the syntactic definition (i.e., the definition on Γ_R) and semantic definition (i.e., the definition on Σ_R) which satisfy the correctness criterion are given. The correctness criterion is characterized by Figure 6.9.

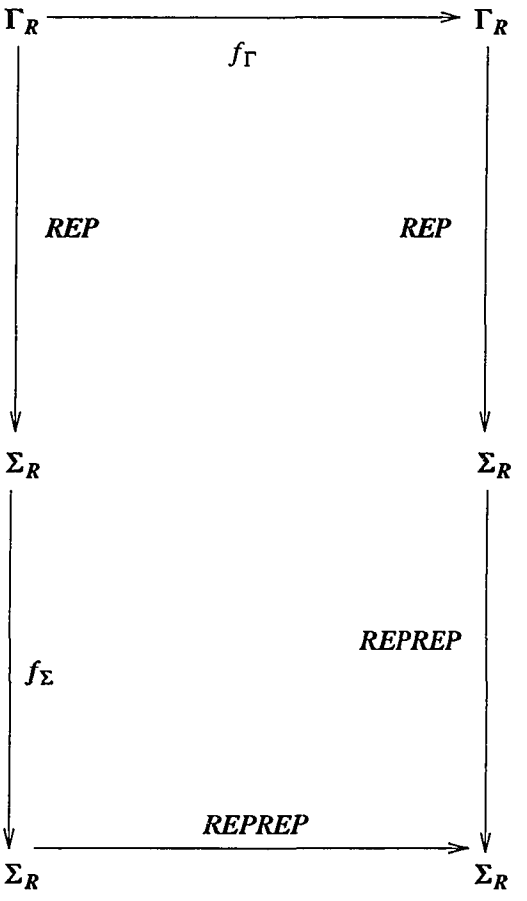


Figure 6.9: The Notion of Correctness

As has been defined earlier, the mapping *REPREP* maps an E-table, T, over scheme R, to elements of Σ_R. $REPREP(T)$ consists a set of definite relations of which exactly one represents the real world truth. In order to extend a relational operator f to E-tables, it must be stipulated that the extended operator captures the effect of the corresponding regular operator on the various definite relations represented in the information content of E-tables. The information content here refers to *REPREP* as *REP* represents the intermediate information content.

Formally, for each operator f, we first need to define f_Σ on Σ_R and then define f_Γ on Γ_R, that satisfies the following conditions:

$REPREP(REP(f_\Gamma(T))) = REPREP(f_\Sigma(REP(T)))$, for unary f, and

$REPREP(REP(f_\Gamma(T_1,T_2))) = REPREP(f_\Sigma(REP(T_1),REP(T_2)))$, for binary f.

6.4.1 Selection

Definition 6.4.1.1 Let R be a relational scheme, $U \in \Sigma_R$, and F be a formula involving operands that are constants of the domains of R or attribute numbers, arithmetic comparison operators $<$, $=$, $>$, $\leq$, $\geq$, $\neq$, and logical operators $\wedge$, $\vee$, and $\neg$, then selection on elements of Σ_R is defined as a mapping $\sigma_F: \Sigma_R \rightarrow \Sigma_R$ such that

$\sigma_F(U) = (U')$, where

$U' = \{r \mid (\exists r_1)(r_1 \in U \wedge (r = \{t \mid (\exists t_1)(t \in r_1 \wedge t \notin \lambda \wedge F(t))\}))\}$

To perform a selection on an E-table, the definite tuples that satisfies the selection conditions should be included in the definite component of the selection as it is. Those definite tuples that do not satisfy the selection conditions should be removed from the definite component. For the tuple sets of the indefinite component, tuples of tuple sets that satisfy the selection conditions should be included in their respective tuple sets and tuples that do not satisfy the selection conditions are to be replaced by a special dummy value and included in their corresponding tuple sets. Formally, selection on E-tables is defined as follows.

Definition 6.4.1.2 Let T be a consistent and reduced E-table over R and F be a formula involving operands that are constants of the domains of R or attribute numbers, arithmetic comparison operators $<, =, >, \leq, \geq, \neq$, and logical operators $\wedge$, $\vee$, and $\neg$, then selection on T is defined as a mapping $\sigma_F : \Gamma_R \rightarrow \Gamma_R$ such that $\sigma_F(T) = REDUCE(T')$, where

$$T'_D = \{t \mid (t \in T_D \wedge F(t))\},$$

$$T'_E = \{W \mid (\exists W_1)(W_1 \in T_E \wedge W = \{w \mid (\exists w_1)(w_1 \in W_1 \wedge w = \{t \mid (\exists t_1)(t_1 \in w_1 \wedge ((F(t_1) \wedge t = t_1) \vee (\neg F(t_1) \wedge t = MAP(t_1)) \vee (t_1 \in \lambda \wedge t = t_1)))\})\})\},$$

where $MAP(t_1, t_2, \ldots, t_n)$ is an n-ary mapping which maps a set of tuple values (including special dummy values) to the elements of λ such that $MAP(t_1, t_2, \ldots, t_n) = MAP(t'_1, t'_2, \ldots, t'_n)$ if and only if $t_1 = t'_1 \wedge t_2 = t'_2 \wedge \cdots \wedge t_n = t'_n$. Examples of the extended selection operator are shown in Figure 6.10 and Figure 6.11.

The correctness of the extended selection operator is established in the following theorem.

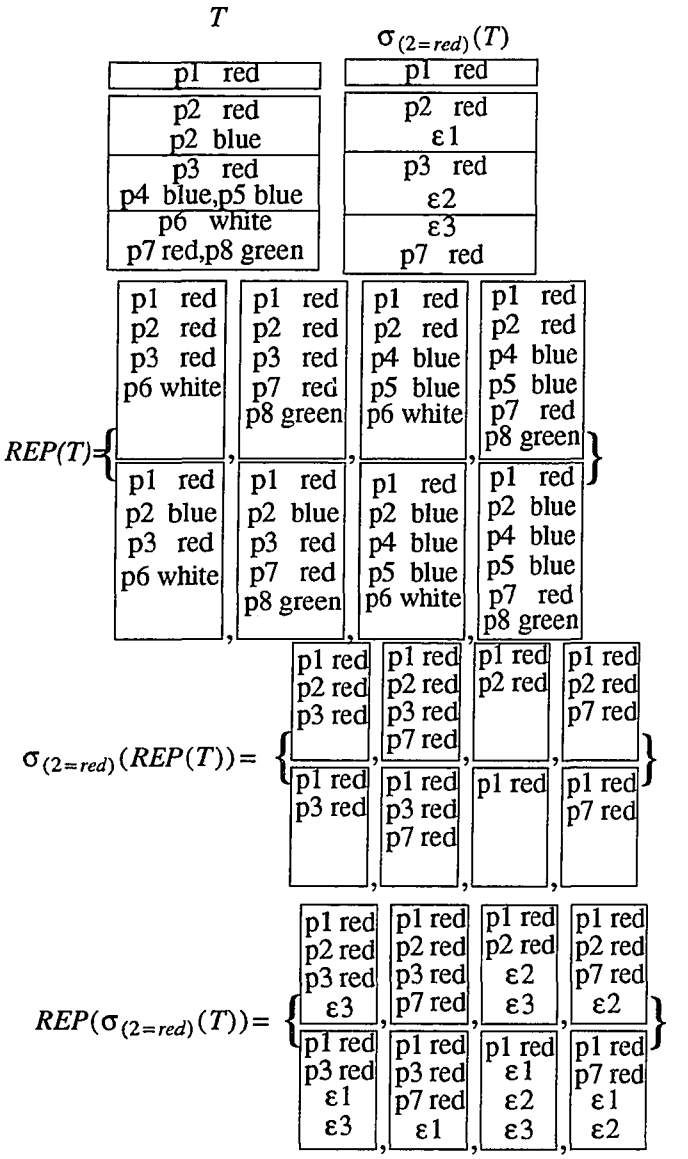
Theorem 6.4.1.1 $REPREP(REP(\sigma_F(T))) = REPREP(\sigma_F(REP(T)))$ for any consistent and reduced E-table T and formula F.

Theorem 6.4.1.1 is also illustrated in Figure 6.10 and Figure 6.11.

6.4.2 Projection

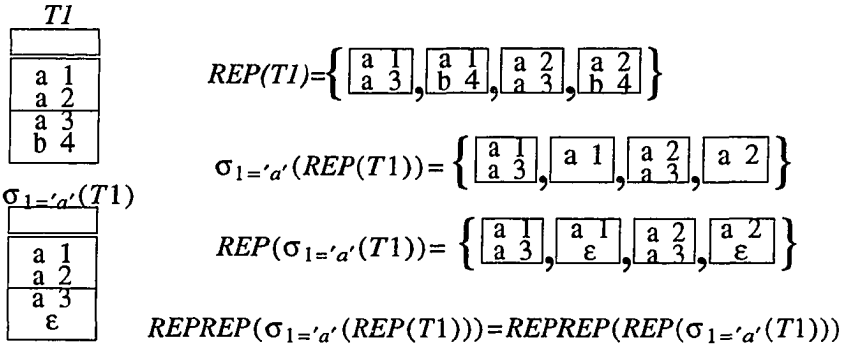
For the definition of projections, the notion of an *accumulation* is necessary and is thereby defined as follows:

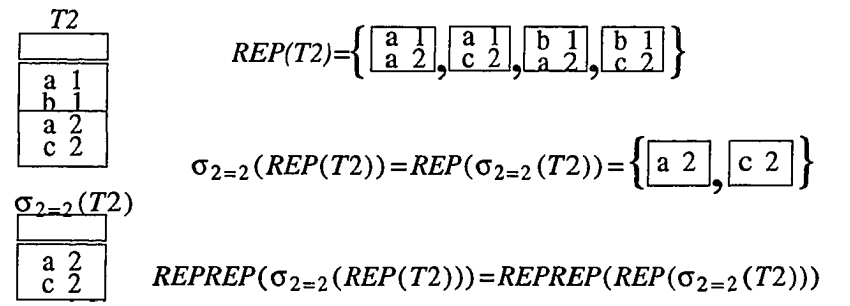
Definition 6.4.2.1 An *accumulation* is a collection of (possibly repeated) elements or objects that satisfy a given condition. An *accumulation* is denoted by a list of elements separated by commas and enclosed in a pair of braces with subscript A at the bottom of the right brace.



$$REPREP(REP(\sigma_{(2=red)}(T)))=REPREP(\sigma_{(2=red)}(REP(T)))$$

Figure 6.10: Selection





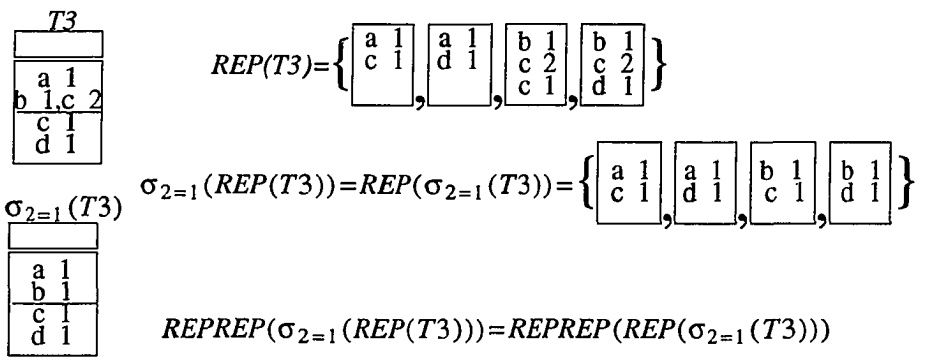


Figure 6.11: Selection

For instance, $S=\{apple,orange,orange,grape\}_A$ is an accumulation. Note that the difference between an accumulation and a set is that multiple appearance of an elements is allowed in an accumulation. For example, *orange* in the accumulation S is listed twice.

Next we extend the equal comparison operator to accumulations.

Definition 6.4.2.2 An accumulation S_1 is equal to another accumulation S_1, denoted by $S_1=S_2$, if every member of S_1 is also a member of S_2 and every member of S_2 is also a member of S_1. Further, for every element of s_1, the number of times it appears in S_1 is the same as the number of times it appears in S_2.

For example, let $S_1=\{1,1,2,3\}$, $S_2=\{1,2,3\}$, and $S_3=\{1,2,3,1\}$. Then $S_1=S_3$ and $S_1\neq S_2$.

Let us now first define projections on Σ_R.

Definition 6.4.2.3 Let R be a relational scheme and $U\in\Sigma_R$ and $A\subseteq R$, then projection on the elements of Σ_R is defined as a mapping $\pi_A:\Sigma_R\rightarrow\Sigma_R$ as follows:

$\pi_A(U)=REDUCEACCACC(CONTRADICTION(U'))$, where

1. U' is a an accumulation of accumulations and is defined as follows:

$$U'=\{r|(\exists r_1)(r_1\in U\wedge r=\{t|(\exists t_1)(t_1\in r_1\wedge((t_1\notin\lambda\wedge t=\pi_A(t_1))\vee(t_1\in\lambda\wedge t=MAP(t_1))))\}_A)\}_A$$

2. *REDUCEACCACC* reduces an accumulation of accumulations U and then converts U to a set of sets (i.e., relations of Σ_R). Note that *REDUCEACCACC* removes duplicated elements from a set of accumulations.

$$REDUCEACCACC(U)=\{SET(r)|(r\in U\wedge\neg((\exists r_1)(r_1\in U\wedge SET(r_1)\subset SET(r)))\wedge\neg((\exists r_1)(r_1\in U\wedge SET(r_1)=SET(r)\text{ and }r_1\neq r)))\},$$

where *SET* is a mapping which converts an accumulation into a set. For example, $SET(\{1,2,2,3\}_A)=\{1,2,3\}$.

The projection on an E-table could mainly cause three complications: (1) redundancies (e.g., $T3$ of Figure 6.13, $T1$ of Figure 6.14, $T1$ and $T2$ of figure 6.15. etc.), (2) entanglements (e.g., $T2$ of Figure 6.13, $T1$ of Figure 6.17, $T2$ of Figure 6.18, etc.), and (3) redundancies mixed with entanglement (e.g., $T1$ of Figure 6.18 and $T1$ of Figure 6.20). The *REDUCEACC* operator is defined to handle the first and second complications, whereas The concept of accumulations together with the *REDUCEACCACC* operator are used to deal with the third complication.

Informally, the projection of the definite component of an E-table is similar to that of the conventional relations. However, some tuple sets of the indefinite component may become singletons on projection in which case they are moved to the definite component. The projection of a special dummy value is the dummy value itself.

Definition 6.4.2.4 Let T be a consistent and reduced E-table over R and $A \subset R$, then projection on E-tables is defined formally as a mapping $\pi_A : \Gamma_R \rightarrow \Gamma_R$ as follows:

$$\pi_A(T) = REDUCE(CONTRADICTION(T'))\text{, where}$$

$$T'_D = \{t \mid (\exists t_1)(t_1 \in T_D \wedge t = t_1[A]) \vee (\exists W)(W \in T_E \wedge (\forall w)(w \in W \rightarrow ((\forall t_1)(t_1 \in w \rightarrow t_1[A] = t))))\},$$

$$T'_E = \{W \mid (\exists W_1)(W_1 \in T_E \wedge W = \{w \mid (\exists w_1)(w_1 \in W_1) \wedge w = \{t \mid (\exists t_1)(t_1 \in w_1 \wedge ((t_1 \notin \lambda \wedge t = t_1[A]) \vee (t_1 \in \lambda \wedge t = t_1)))\}\} \wedge |W| > 1)\},$$

The correctness of the extended projection operator is established in the following theorem.

Theorem 6.4.2.1 $REPREP(REP(\pi_A(T))) = REPREP(\pi_A(REP(T)))$ for any consistent and reduced domain compatible E-tables T.

Examples of projection and Theorem 6.4.2.1 are illustrated in Figure 6.12 through Figure 6.20.

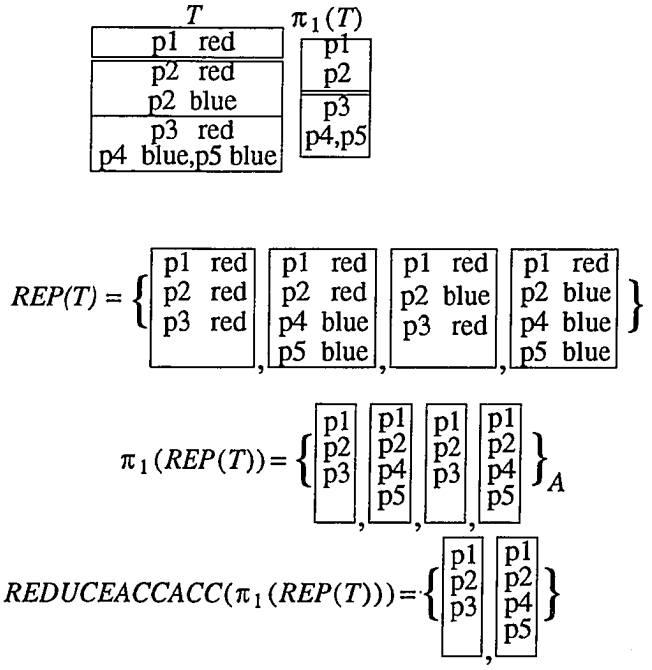
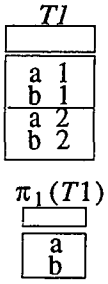


Figure 6.12: Projection

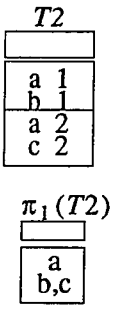


$$REP(T1)=\left\{\begin{bmatrix}a&1\\a&2\end{bmatrix},\begin{bmatrix}a&1\\b&2\end{bmatrix},\begin{bmatrix}b&1\\a&2\end{bmatrix},\begin{bmatrix}b&1\\b&2\end{bmatrix}\right\}$$

$$\pi_1(REP(T1))=\left\{\begin{bmatrix}a\\a\end{bmatrix},\begin{bmatrix}a\\b\end{bmatrix},\begin{bmatrix}b\\a\end{bmatrix},\begin{bmatrix}b\\b\end{bmatrix}\right\}_A$$

$$REDUCEACCACC(\pi_1(REP(T)))=\left\{\begin{bmatrix}a\end{bmatrix},\begin{bmatrix}b\end{bmatrix}\right\}$$

$$REPREP(REDUCEACCACC(\pi_1(REP(T1)))=REPREP(REP(\pi_1(T1)))$$

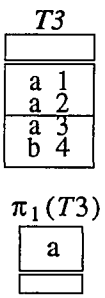


$$REP(T2)=\left\{\begin{bmatrix}a&1\\a&2\end{bmatrix},\begin{bmatrix}a&1\\c&2\end{bmatrix},\begin{bmatrix}b&1\\a&2\end{bmatrix},\begin{bmatrix}b&1\\c&2\end{bmatrix}\right\}$$

$$\pi_1(REP(T2))=\left\{\begin{bmatrix}a\\a\end{bmatrix},\begin{bmatrix}a\\c\end{bmatrix},\begin{bmatrix}b\\a\end{bmatrix},\begin{bmatrix}b\\c\end{bmatrix}\right\}_A$$

$$REDUCEACCACC(\pi_1(REP(T2)))=\left\{\begin{bmatrix}a\end{bmatrix},\begin{bmatrix}b\\c\end{bmatrix}\right\}$$

$$REPREP(REDUCEACCACC(\pi_1(REP(T2)))=REPREP(REP(\pi_1(T2)))$$



$$REP(T3)=\left\{\begin{bmatrix}a&1\\a&3\end{bmatrix},\begin{bmatrix}a&1\\b&4\end{bmatrix},\begin{bmatrix}a&2\\a&3\end{bmatrix},\begin{bmatrix}a&2\\b&4\end{bmatrix}\right\}$$

$$\pi_1(REP(T3))=\left\{\begin{bmatrix}a\\a\end{bmatrix},\begin{bmatrix}a\\b\end{bmatrix},\begin{bmatrix}a\\a\end{bmatrix},\begin{bmatrix}a\\b\end{bmatrix}\right\}_A$$

$$REDUCEACCACC(\pi_1(REP(T3)))=\left\{\begin{bmatrix}a\end{bmatrix}\right\}$$

$$REPREP(REDUCEACCACC(\pi_1(REP(T3)))=REPREP(REP(\pi_1(T3)))$$

Figure 6.13: Projection

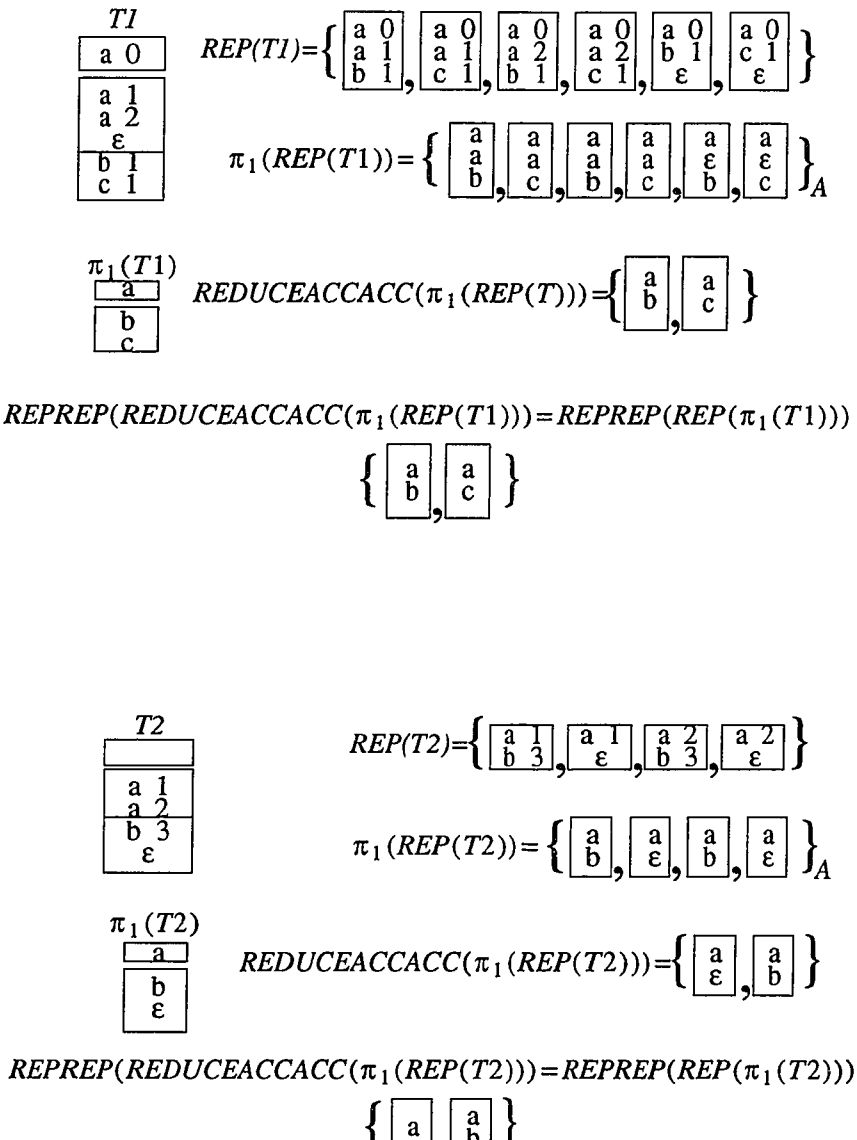
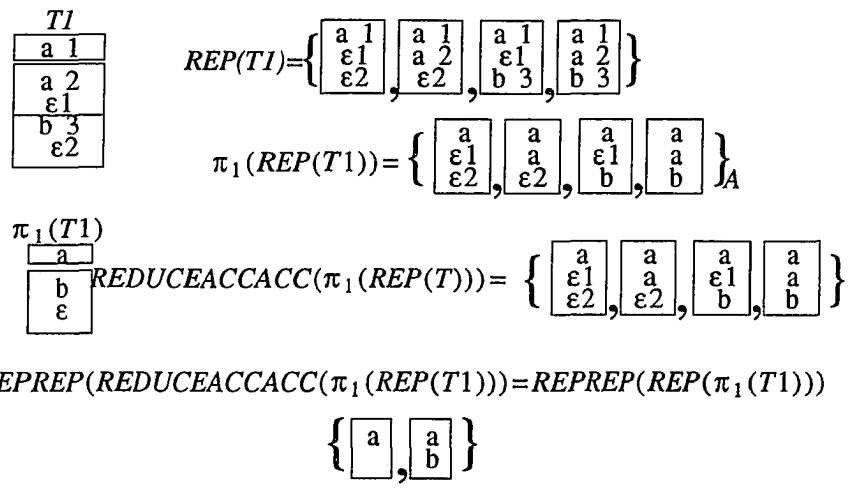


Figure 6.14: Projection



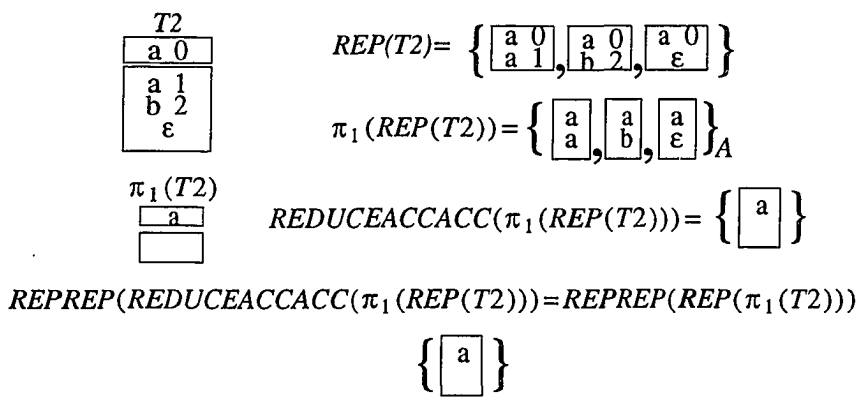


Figure 6.15: Projection

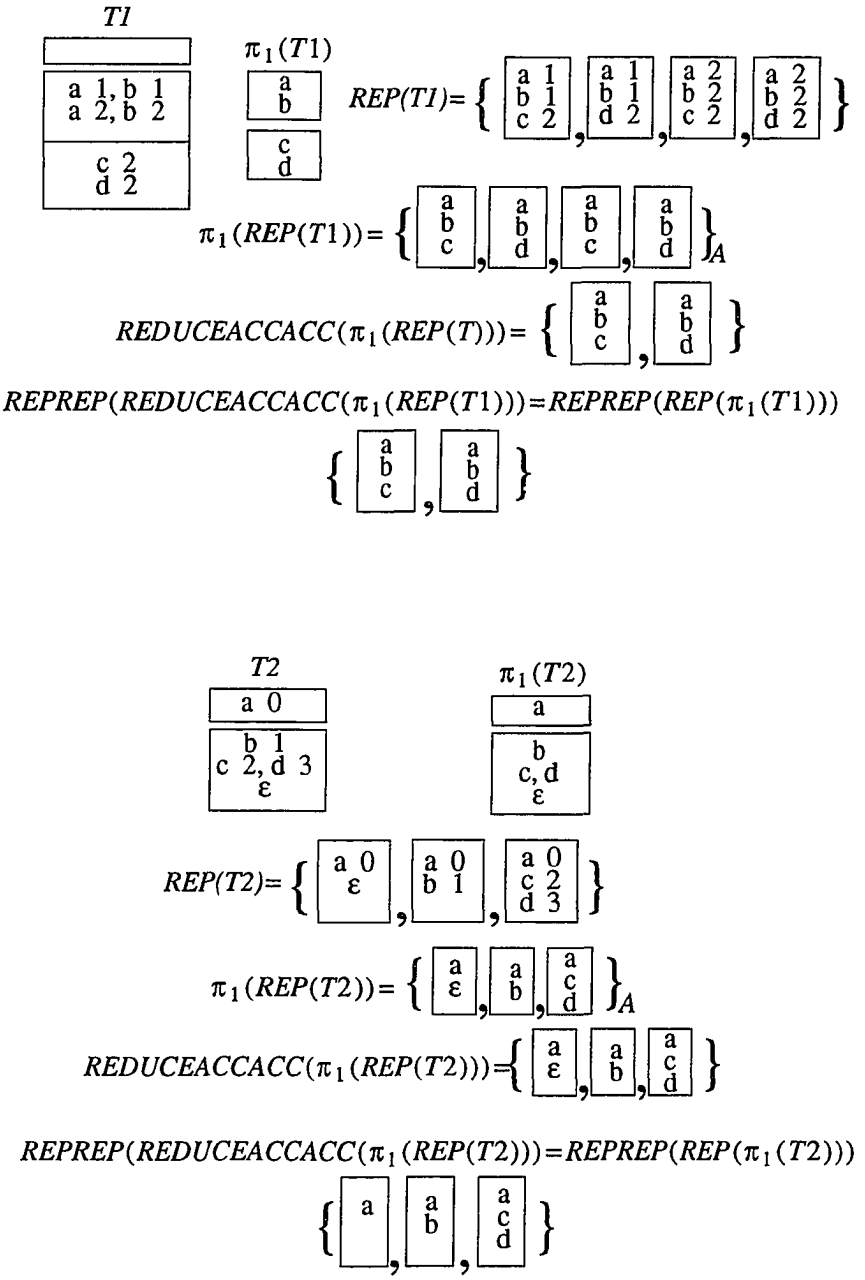
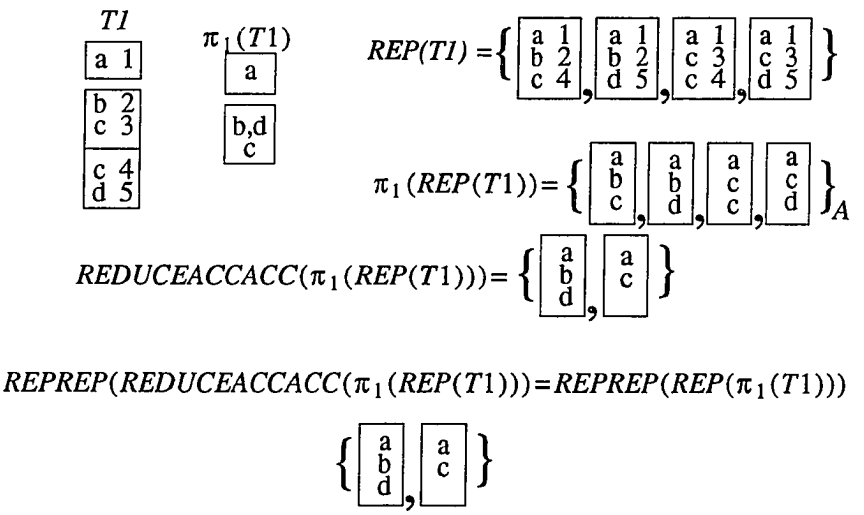


Figure 6.16: Projection



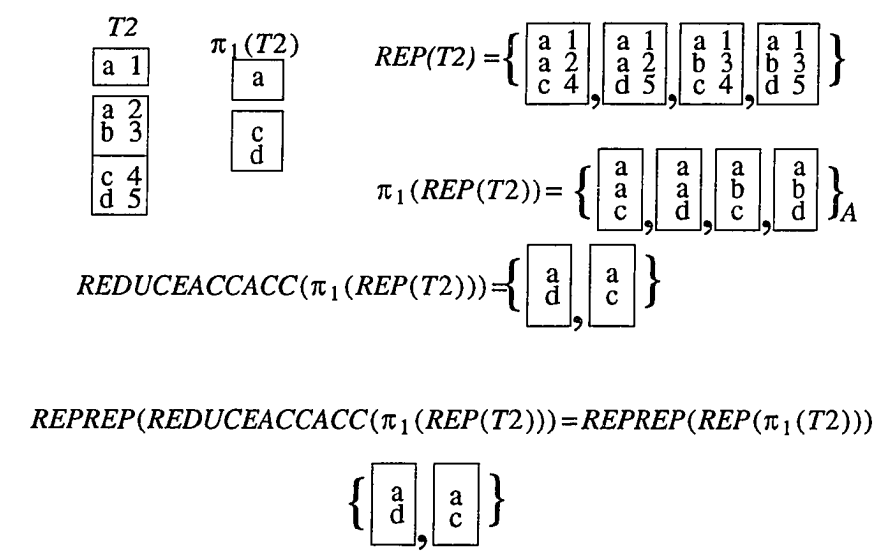
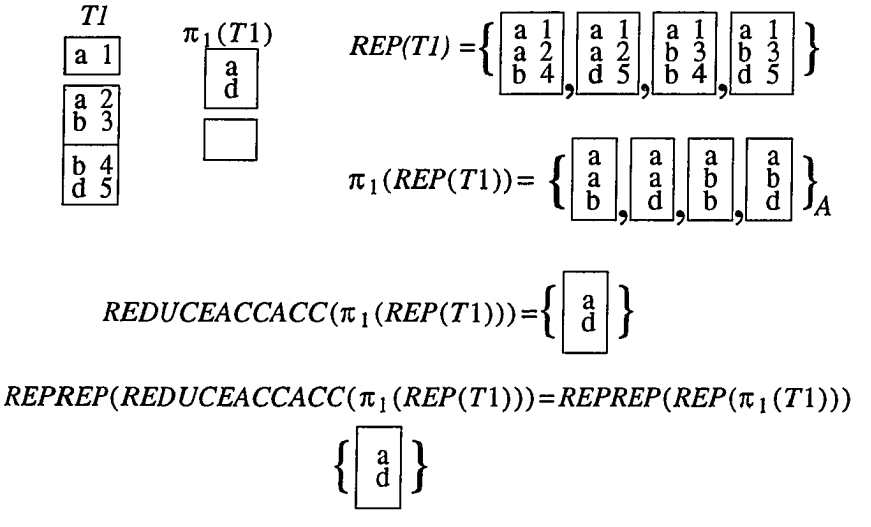


Figure 6.17: Projection



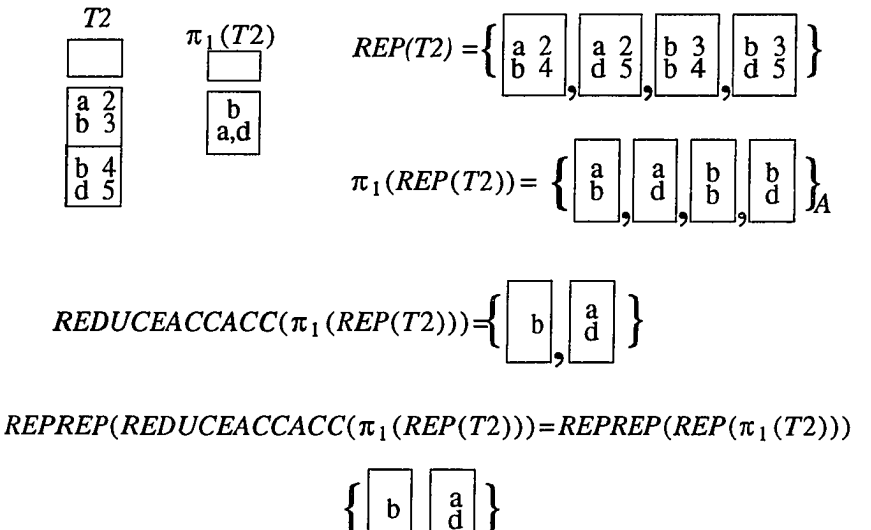


Figure 6.18: Projection

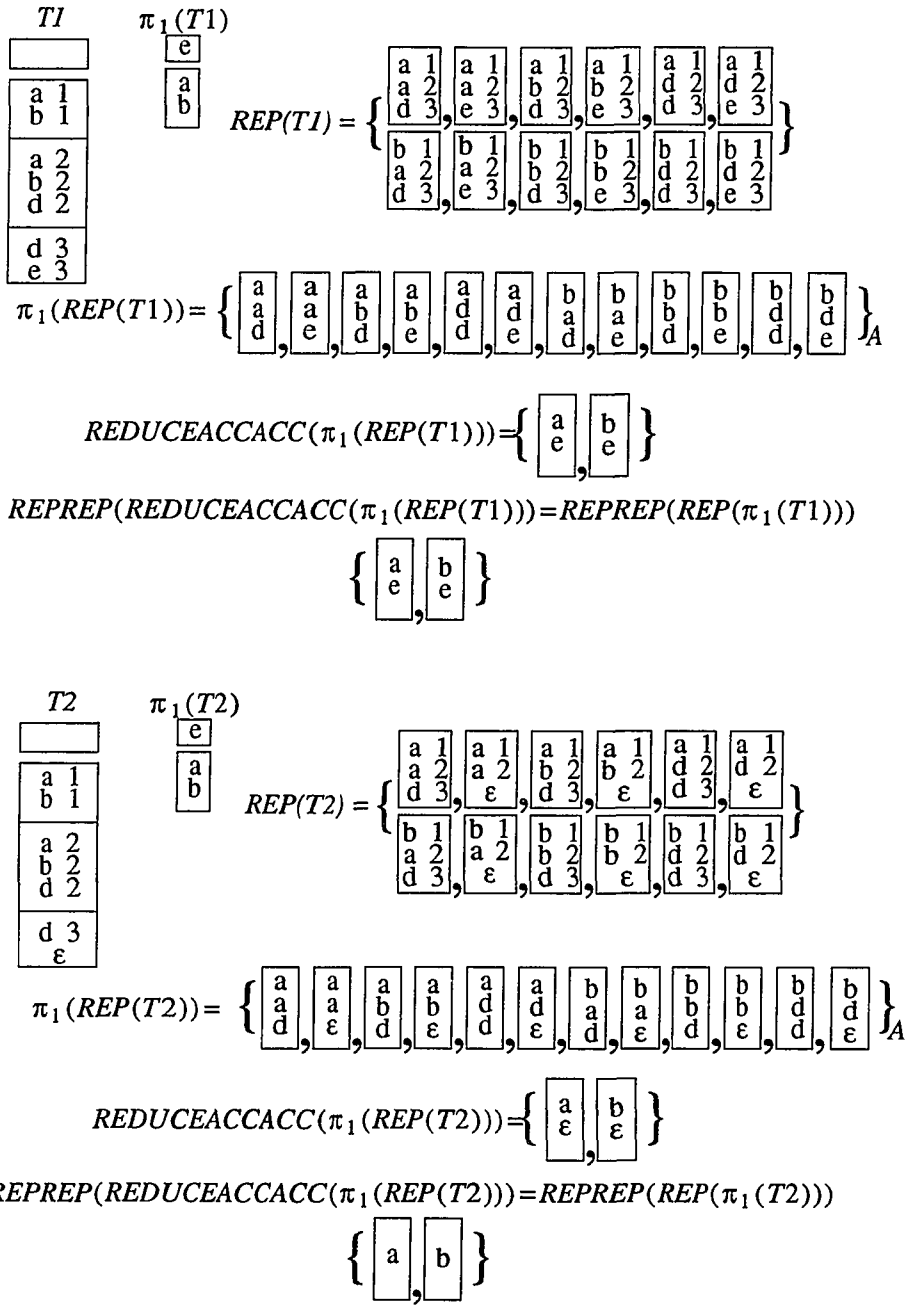


Figure 6.19: Projection

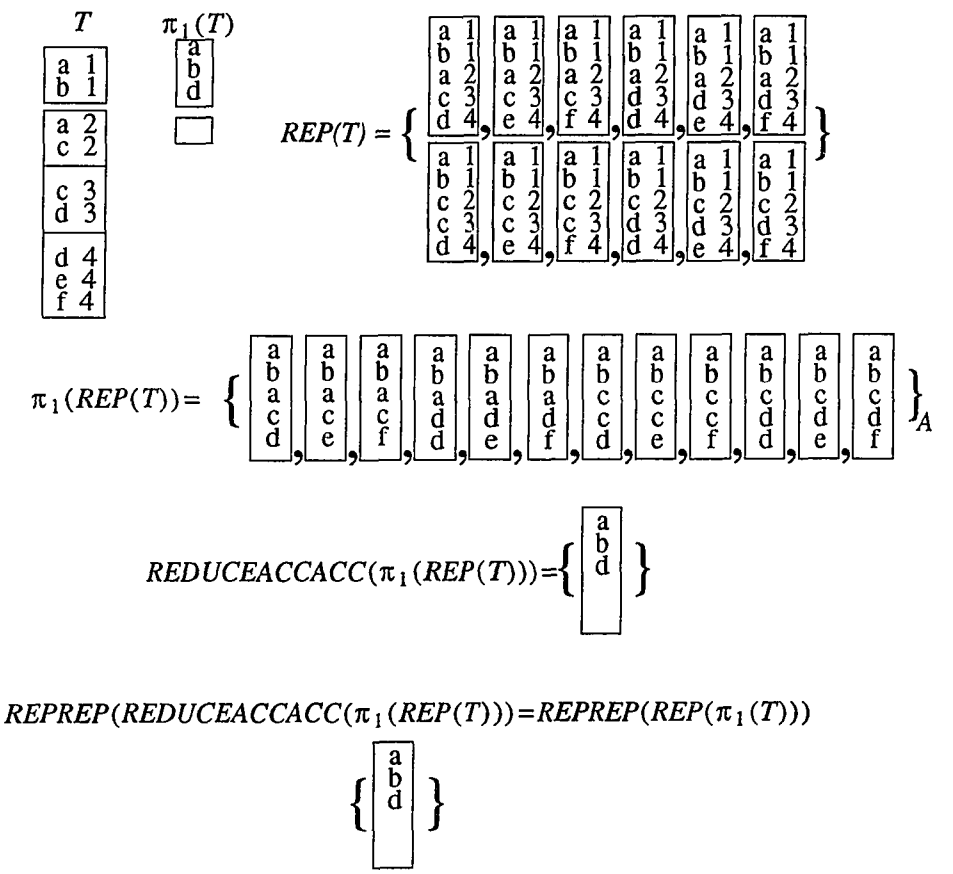


Figure 6.20: Projection

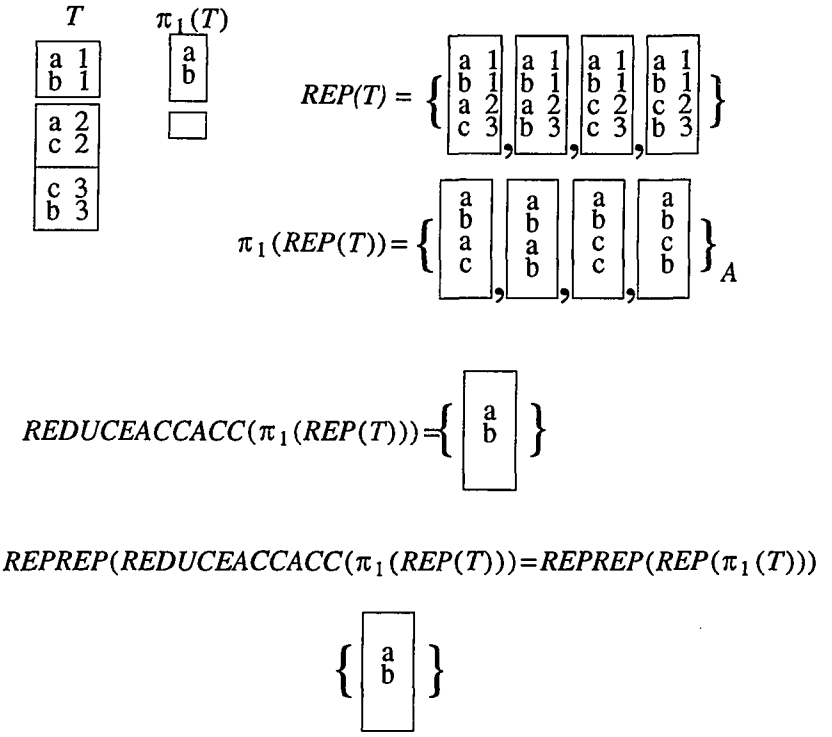


Figure 6.21: Projection

6.4.3 Cartisian product

Definition 6.4.3.1 The cartisian product on the elements of Σ_R is a mapping $\times:\Sigma_R\times\Sigma_R\rightarrow\Sigma_R$. Let U_1 and U_2 be members of Σ_R, then

$$U_1\times U_2 = REDUCEREP(CONTRADICTION(U)), \text{ where}$$

$$U=\{r|(\exists r_1)(\exists r_2)(r_1\in U_1 \wedge r_2\in U_2 \wedge$$

$$r=\{t|(\exists t_1)(\exists t_2)(t_1\in r_1 \wedge t_2\in_2 \wedge t_1\notin\lambda \wedge t_2\notin\lambda \wedge t=t_1.t_2)\})\}$$

Cartisian products under the E-table model is quite straightforward. Consider the example in Figure 6.21, the cartisian product of the two definite components constitutes the definite component. The indefinite component of the cartisian product is obtained in the following

manner: if $a2$ of the set of tuple sets of the indefinite component of $T1$ and $b3$ of the set of the tuple sets of the indefinite component of $T2$ were true, then taking the cartisian product between $a2$ and $T2_D$, $a2$ and $b3$, and $T1_D$ and $b3$ forms the tuple set $\{a2b1, a2b2, a2b3, a1b3\}$, which should be included in a set of tuple sets of the indefinite component. The rest of the tuple sets can be obtained by exhausting all the possible truth combinations of the outcomes of $T1_E$ and $T2_E$ (e.g., $a2$ and $b4$ were true, $a2$ and $b5$ were true, $a3$ and $b3$ were true, etc.).

Formally, the cartisian product of E-tables is defined as a mapping $\times: \Gamma_{R_1} \times \Gamma_{R_2} \rightarrow \Gamma_{R_1.R_2}$ as follows.

Definition 6.4.3.2: Let T_1 and T_2 be consistent and reduced E-tables under relational schemes R_1 and R_2 such that $T_E^1 = \{W_1^1, \ldots, W_m^1\}$ and $T_E^2 = \{W_1^2, \ldots, W_n^2\}$. Let

$$E = \{(w_1 \cup \cdots \cup w_m) - (T_D^1 \cup \lambda) \mid (\forall i)(1 \le i \le m \rightarrow w_i \in W_i^1) \wedge (\forall i)(1 \le i \le n \rightarrow w_i \in W_i) \wedge \neg[(\exists u)(u \in \bigcup_{i=1}^{n} (W_i - w_i) \wedge (u \subseteq W)]\}$$

$$F = \{(w_1 \cup \cdots \cup w_n) - (T_D^2 \cup \lambda) \mid (\forall i)(1 \le i \le n \rightarrow w_i \in W_i^2) \wedge (\forall i)(1 \le i \le n \rightarrow w_i \in W_i) \wedge \neg[(\exists u)(u \in \bigcup_{i=1}^{n} (W_i - w_i) \wedge (u \subseteq W)]\}$$

Let the elements of E be E_1, ..., E_e and those of F be F_1, ..., F_f, then $T_1 \times T_2 = REDUCE(CONTRADICTION(T))$, where T is defined as follows:

$$T_D = \{t \mid (\exists t_1)(\exists t_2)(t_1 \in T_D^1 \wedge t_2 \in T_D^2 \wedge t = t_1.t_2)\}$$

$$T_E = \{t \mid (\exists t_1)(\exists t_2)((t_1 \in T_D^1 \wedge t_2 \in F_l) \vee (t_1 \in E_k \wedge t_2 \in T_D^2) \vee (t_1 \in E_k \wedge t_2 \in F_l)) \wedge t_1 \notin \lambda \wedge t_2 \notin \lambda \wedge t = t_1.t_2\}$$

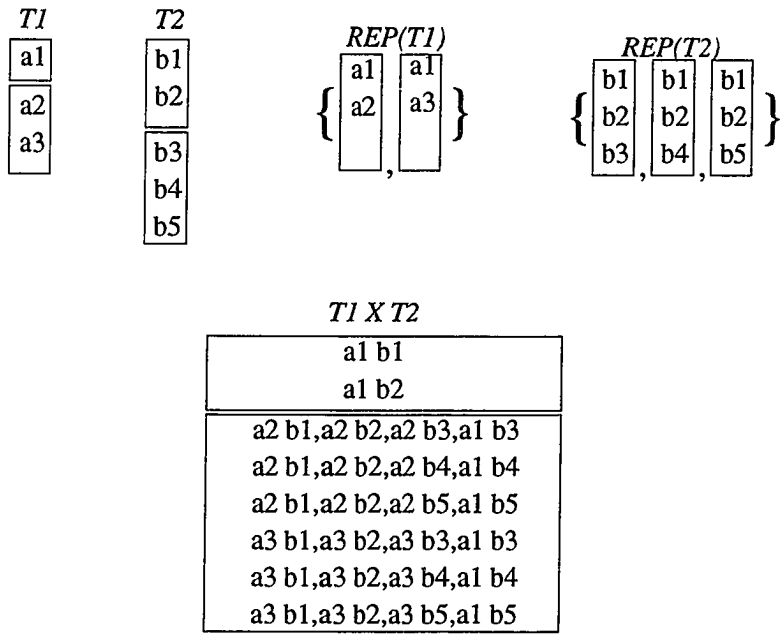


Figure 6.22: Cartisian Product

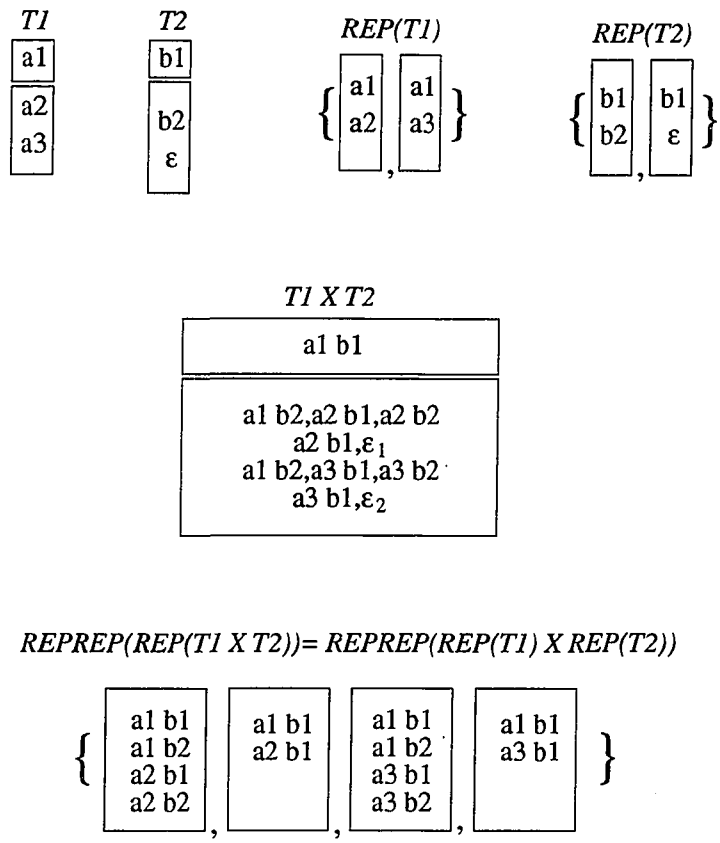


Figure 6.23: Cartisian Product

where $1 \le k \le e$, $1 \le l \le f$, $i = k$ if $e \neq 0$ otherwise $i = 0$, and $j = l$ if $f \neq 0$ otherwise $j = 0$. Example of the extended cartisian product is shown in Figure 6.22 and Figure 6.23.

The correctness of the extended cartisian product is established in the following theorem and is also illustrated in Figure 6.22 and Figure 6.23.

Theorem 6.4.3.1 $REPREP(REP(T_1 \times T_2)) = REPREP(REP(T_1) \times REP(T_2))$ for any consistent and reduced E-tables T_1 and T_2.

6.4.4 Difference

Definition 6.4.4.1 The difference on the elements of Σ_R is defined as a mapping $-:\Sigma_R\times\Sigma_R\rightarrow\Sigma_R$. Let U_1 and U_2 be members of Σ_R, then

$$U_1-U_2=\{r|(\exists r_1)(r_1\in U_1\wedge r=\{t|(t\in(r_1-(\bigcup_{r_2\in U_2} r_2)))\})\}$$

The difference operator on E-tables is defined on E-tables as follows: consider two consistent and reduced domain compatible E-tables T_1 and T_2 and let $T=T_1-T_2$, then:

(1) A tuple t of T_D^1 should be included if t is in neither the definite component of T_2 nor any of the tuple sets of the sets of the tuple sets of the indefinite component of T_2. Otherwise, it is replaced by a special dummy value. That is, t will not be in the result of the difference if there is a possibility that it is true under T_2.

(2) A tuple t of a tuple set w of a set of tuple sets W of T_E^1 should remain in w if it is in neither the definite component of T_2 nor any of the tuple sets of the indefinite component of T_2. Otherwise, t should be replaced by a special dummy value.

The following definition formally defines the difference on E-tables as a mapping $-:\Gamma_R\times\Gamma_R\rightarrow\Gamma_R$.

Definition 6.4.4.2 Let T_1 and T_2 be consistent and reduced domain compatible E-tables, then $T_1-T_2=REDUCE(T)$, where

$$T_D=\{t|(t\in T_D^1\wedge t\notin T_D^2\wedge\neg(\exists W)(\exists w)(W\in T_E^2\wedge w\in W\wedge t\in w))\}$$

$$T_E=\{W|(\exists W_1)(W_1\in T_E^1\wedge W=\{w|(\exists w_1)(w_1\in W_1\wedge$$
$$w=\{t|(\exists t_1)((t_1\in w_1\wedge t_1\notin\lambda\wedge t_1\notin T_D^2\wedge\neg(\exists w_2)(w_2\in T_E^2\wedge t_1\in w_2\wedge t=t_1))\vee$$
$$((t_1\notin\lambda\wedge t_1\in T_D^2)\vee(\exists w_2)(w_2\in T_E^2\wedge t_1\in w_2))\wedge t=MAP(t_1))\vee$$
$$(t_1\in\lambda\wedge t=t_1))\})\})\}$$

Examples of the extended difference operator are shown in Figure 6.24, Figure 6.25, and Figure 6.26.

The correctness of the difference operator is established in the following theorem and is also illustrated by Figure 6.23, and Figure 6.24.

Theorem 6.4.4.1 $REPREP(REP(T_1 - T_2)) = REPREP(REP(T_1) - REP(T_2))$ for any consistent and reduced domain compatible E-tables T_1 and T_2.

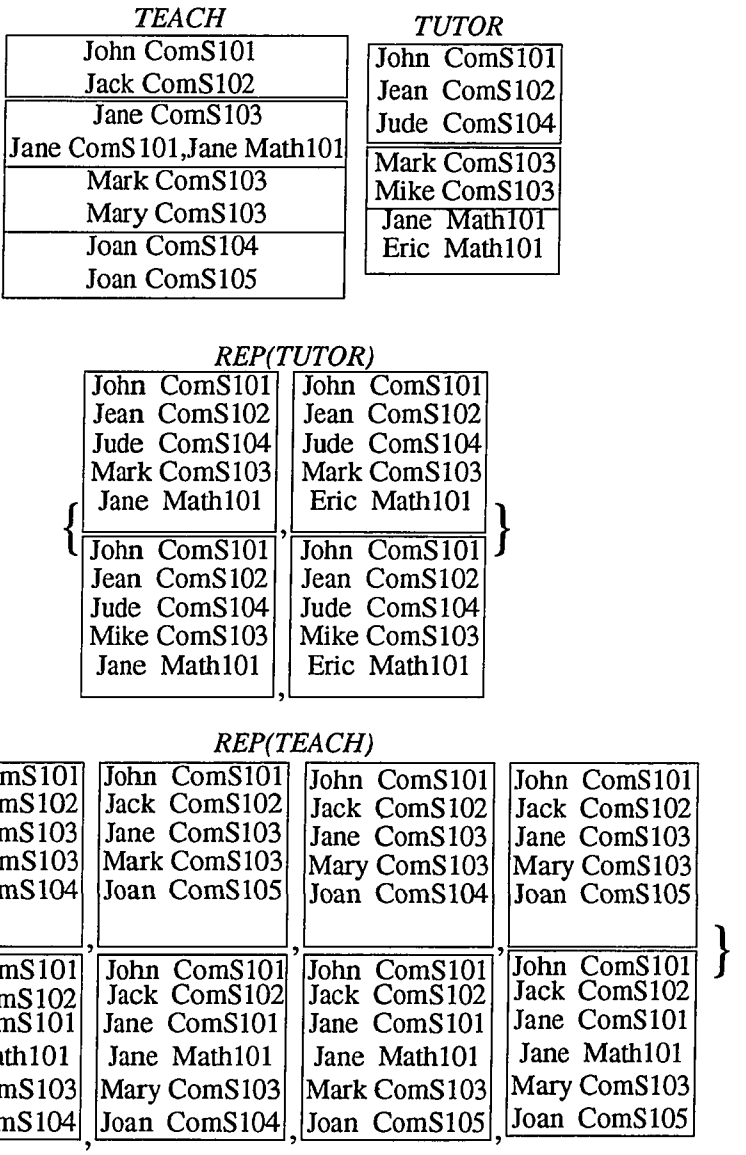


Figure 6.24: Difference

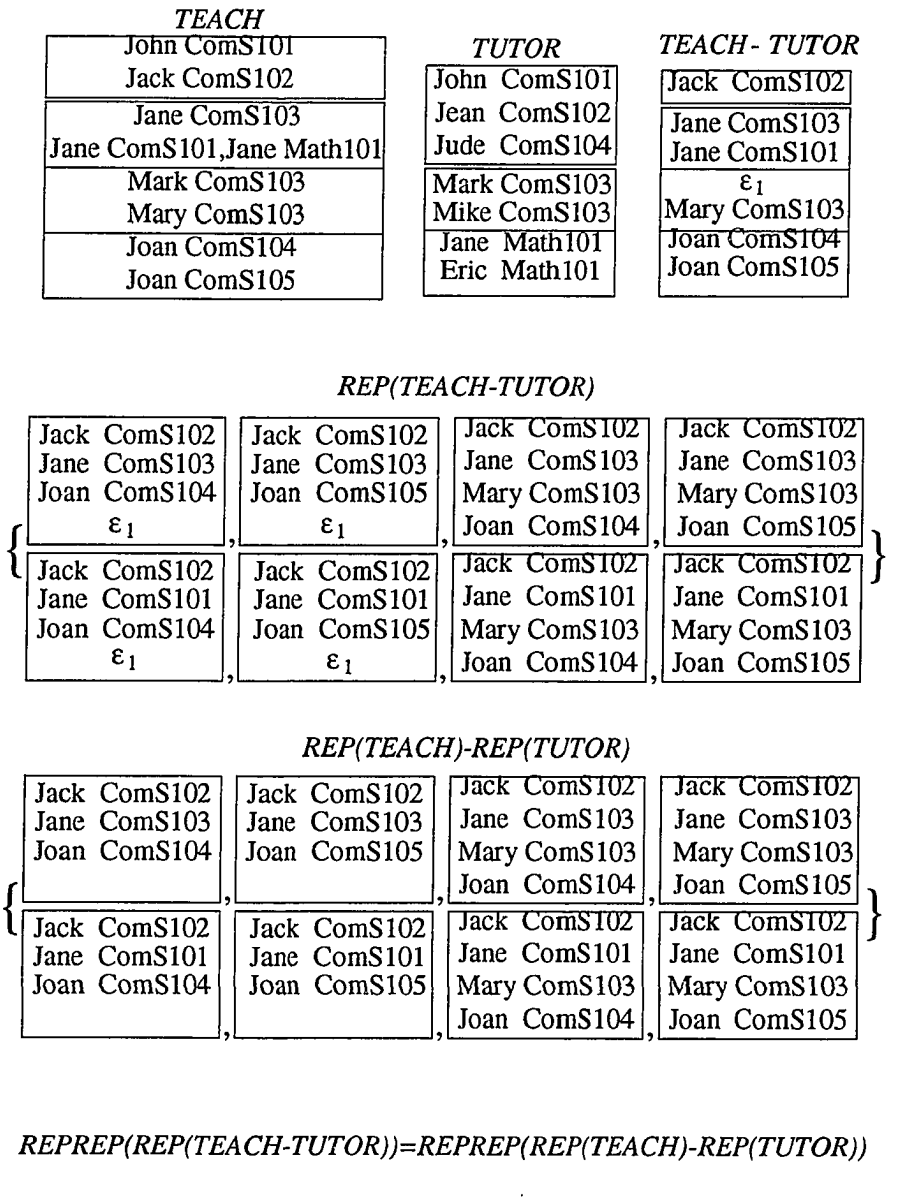


Figure 6.24: Continued

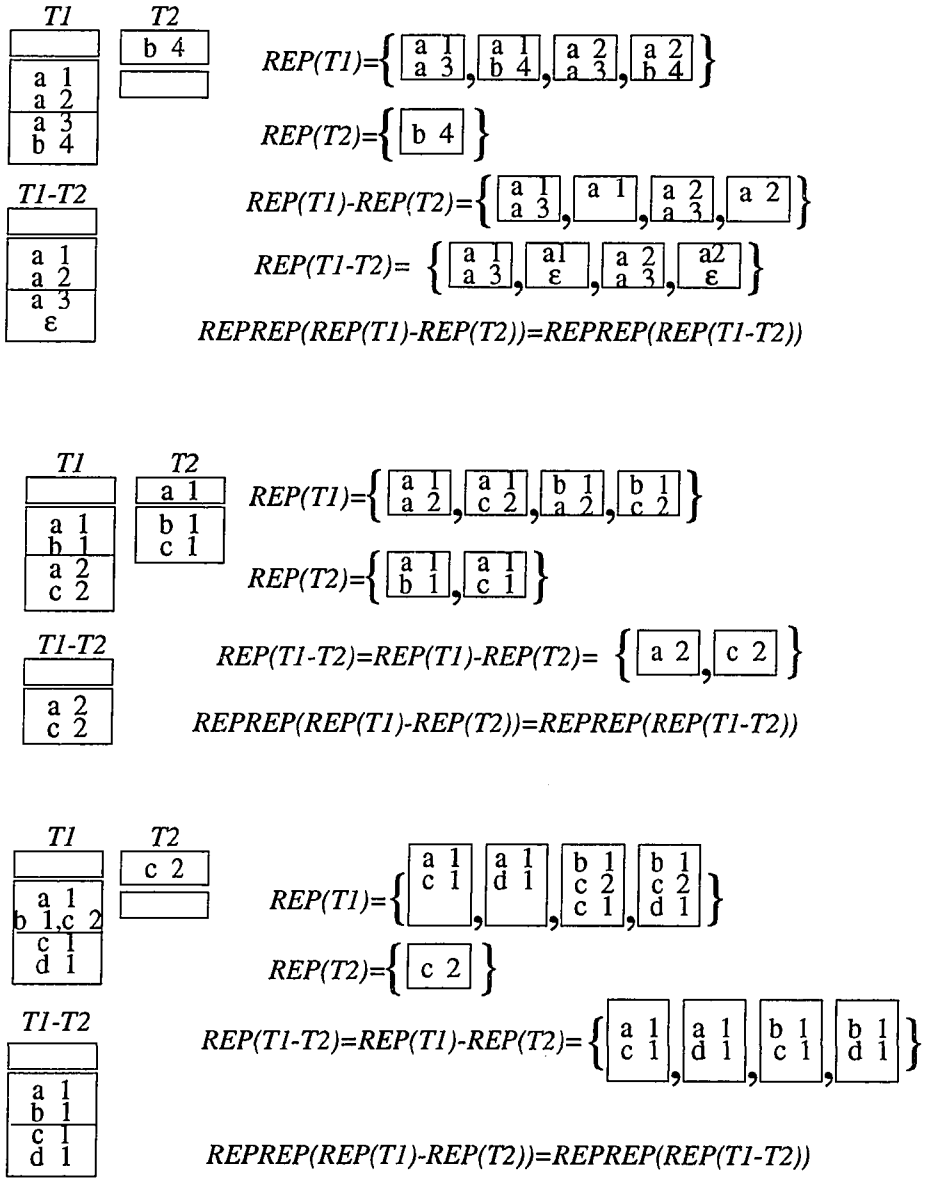


Figure 6.25: Difference

6.4.5 Union

The syntactic definition of the union operator on E-tables is quite simple and is just the union of the elements of the corresponding components. However, the semantic definition is quite complex. The complication is due to the fact that elements of the two different relations, once were independent from each other, could interact with each other after the union. For example, suppose E-table $T1$ contains $\{\{a\},\{b\}\}$ as its indefinite member and E-table $T2$ contains $\{\{a\},\{c\}\}$ as its indefinite member. Once these two elements are merged, the relationship (which does not exist before the union) that if a is true in $(a \mid b)$, then c cannot be true in $(a \mid c)$ emerges. Specifically, redundancies of the first through the fourth and the ninth types identified in Section i6.3 could arise in the union of two reduced and consistent E-tables. Similar to the extended projection operator, the concept of accumulations and the *REDUCEACC* operator are necessary to define the union operator on Σ_R correctly.

Definition 6.4.5.1: Let R be a relational scheme, and U_1 and U_2 be members of Σ_R, then

$U_1 \cup U_2 = REDUCEACC(CONTRADICTION(U))$, where

1. U is an accumulation of relations obtained as follows:

$$U = \{r \mid (\exists r_1)(\exists r_2)(r_1 \in U_1 \wedge r_2 \in U_2 \wedge r = r_1 \cup r_2)\}_A.$$

2. *REDUCEACC* reduces an accumulation of relations U and then converts U to a set of sets (i.e., relations of Σ_R).

$$REDUCEACC(U) = \{r \mid (r \in U \wedge \neg(\exists r_1)(r_1 \in U \wedge r_1 \subseteq r))\}$$

The union of two E-tables is the union of the corresponding components of the two operands. Any redundancies introduced are removed by the *REDUCE* operator. Formally, union is defined as a mapping $\cup : \Gamma_R \times \Gamma_R \rightarrow \Gamma_R$ as follows.

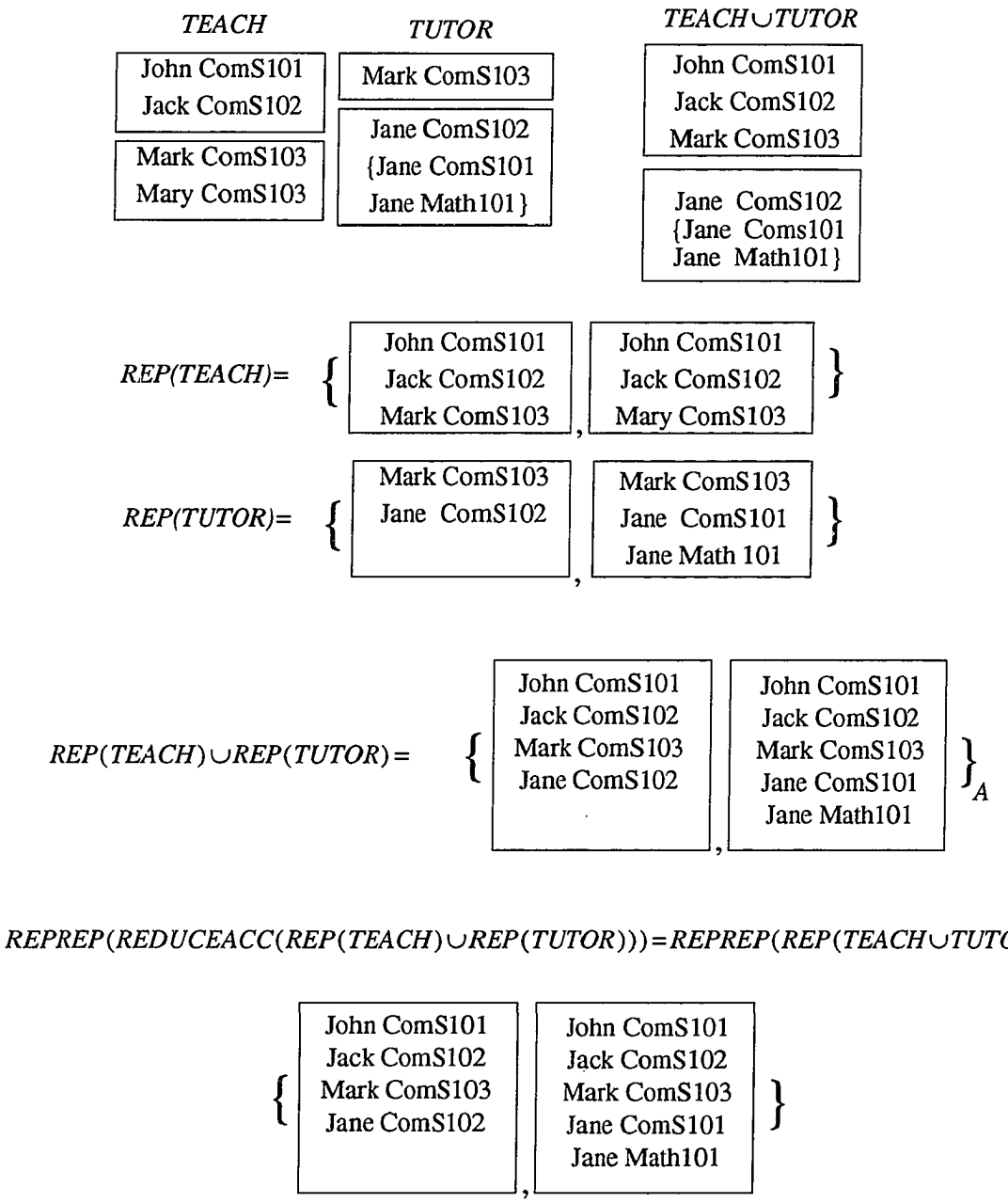
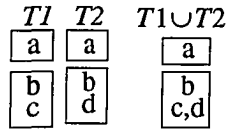


Figure 6.26: Union



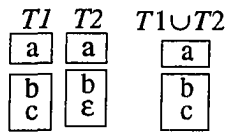
$$REP(T1)=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\c\end{matrix}\right\}\quad REP(T2)=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\d\end{matrix}\right\}$$

$$REP(T1)\cup REP(T2)=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\b\\d\end{matrix},\begin{matrix}a\\b\\c\end{matrix},\begin{matrix}a\\c\\d\end{matrix}\right\}_A$$

$$REDUCEACC(REP(T1)\cup REP(T2))=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\c\\d\end{matrix}\right\}$$

$$REPREP(REDUCEACC(REP(T1)\cup REP(T2)))=REPREP(REP(T1\cup T2))$$

$$\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\c\\d\end{matrix}\right\}$$



$$REP(T1)=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\c\end{matrix}\right\}\quad REP(T2)=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\\varepsilon\end{matrix}\right\}$$

$$REP(T1)\cup REP(T2)=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\b\\\varepsilon\end{matrix},\begin{matrix}a\\b\\c\end{matrix},\begin{matrix}a\\c\\\varepsilon\end{matrix}\right\}_A$$

$$REDUCEACC(REP(T1)\cup REP(T2))=\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\c\\\varepsilon\end{matrix}\right\}$$

$$REPREP(REDUCEACC(REP(T1)\cup REP(T2)))=REPREP(REP(T1\cup T2))$$

$$\left\{\begin{matrix}a\\b\end{matrix},\begin{matrix}a\\c\end{matrix}\right\}$$

Figure 6.27: Union

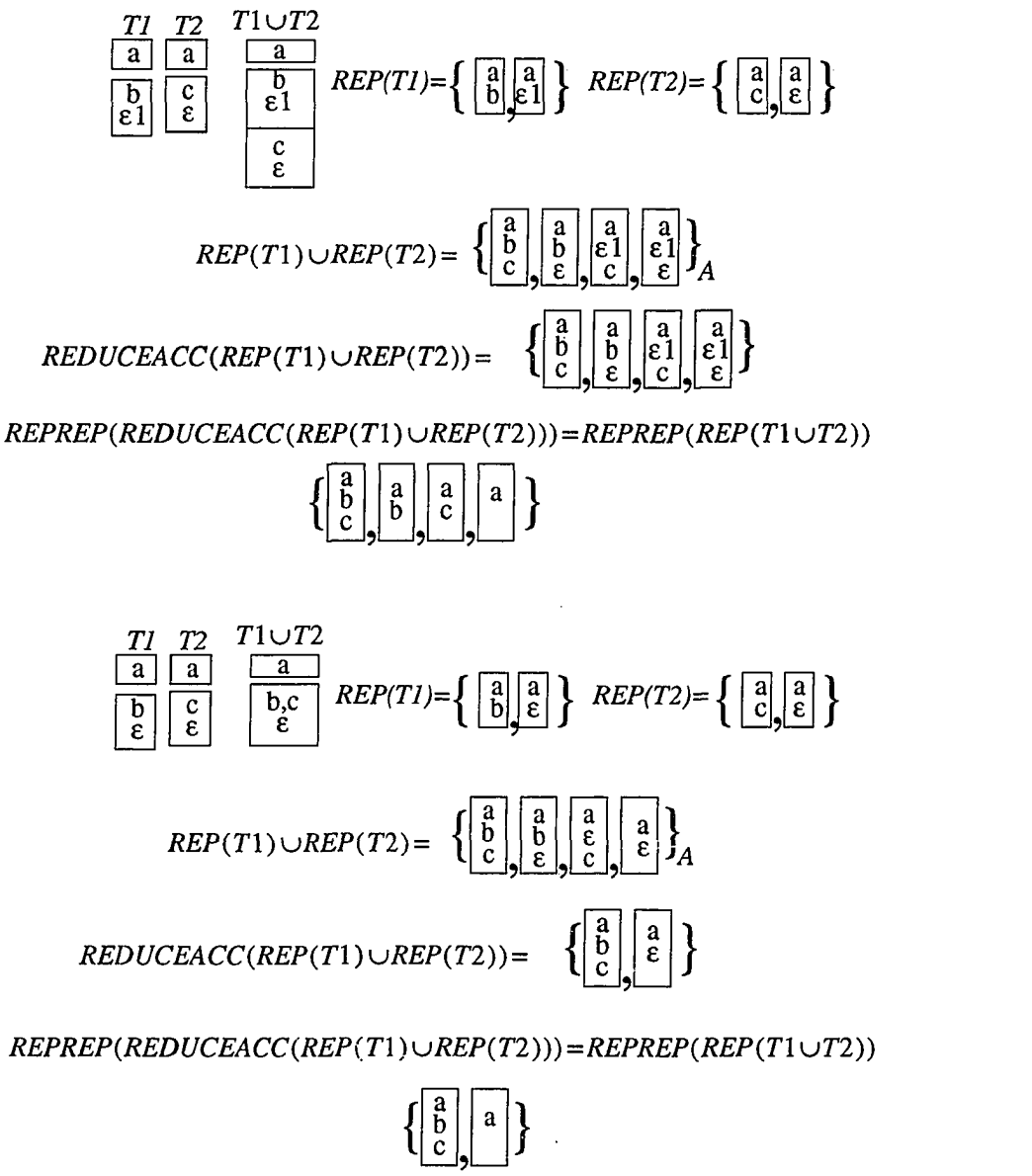


Figure 6.28: Union

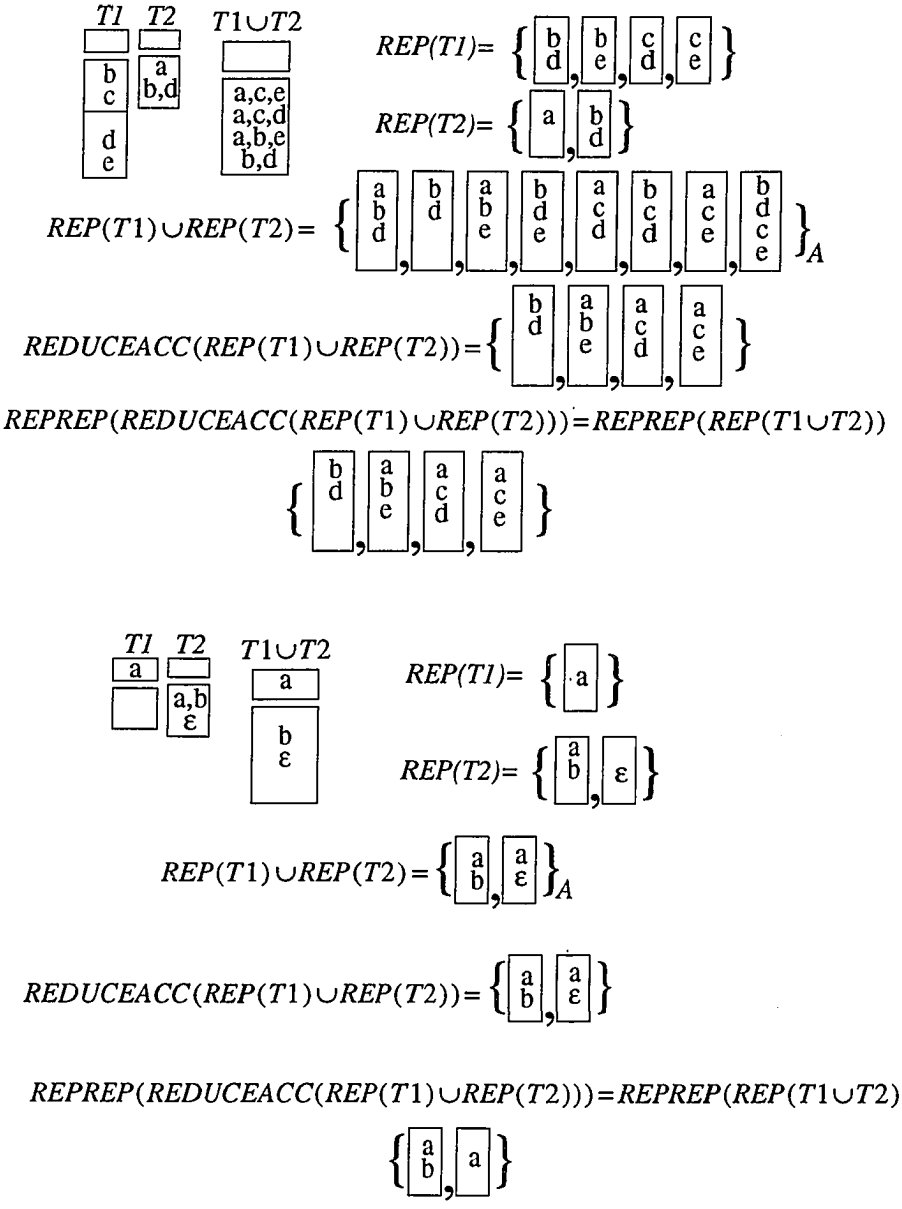


Figure 6.29: Union

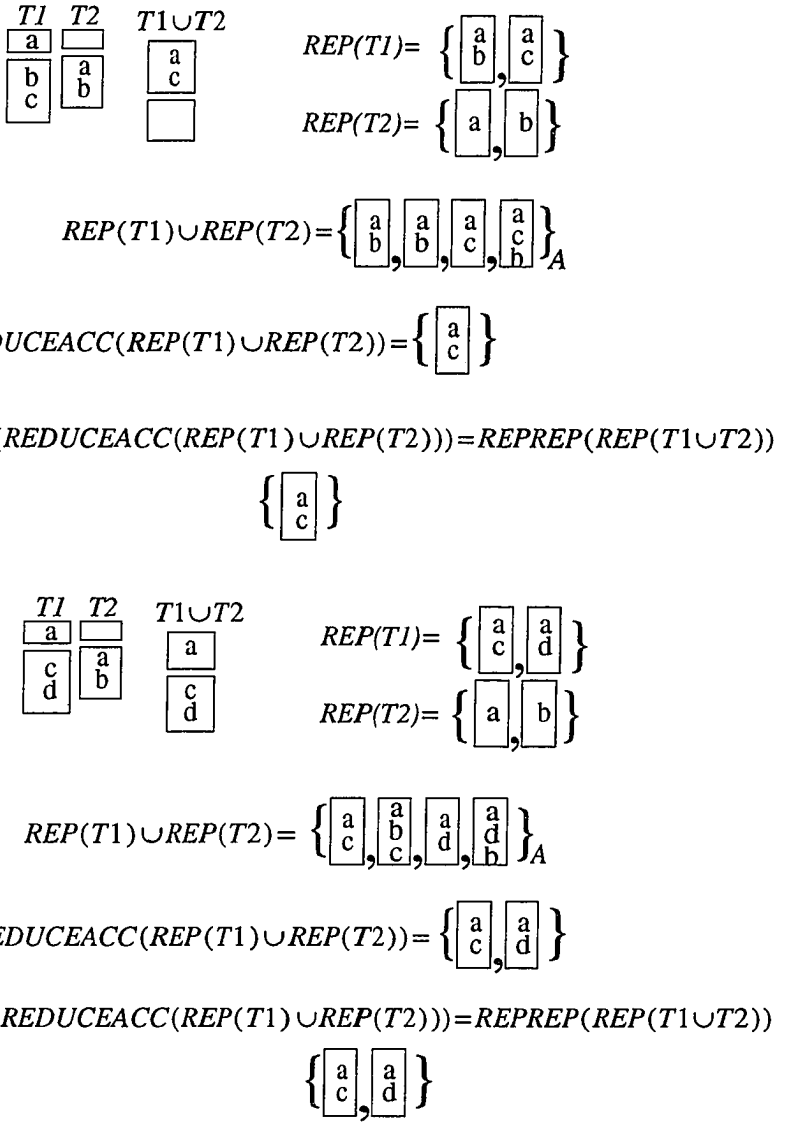


Figure 6.30: Union

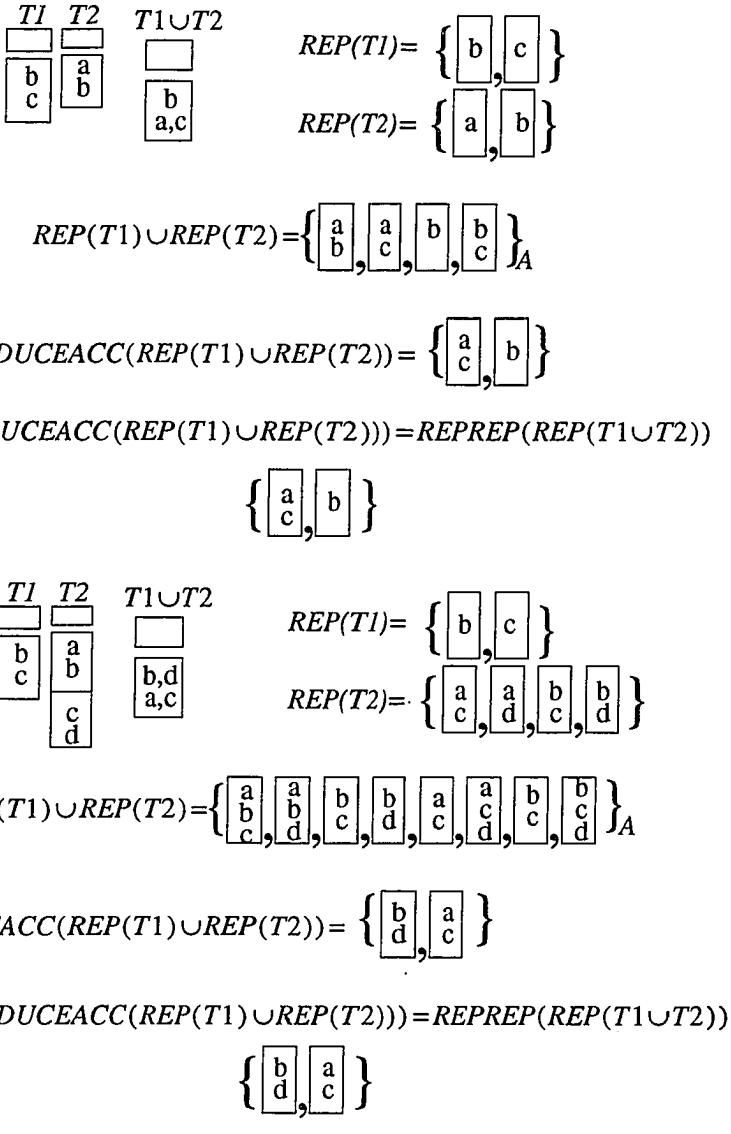


Figure 6.31: Union

Definition 6.4.5.2 Let T_1 and T_2 be consistent and reduced domain compatible E-tables, then

$T_1 \cup T_2 = REDUCE(CONTRADICTION(T))$, where

$T_D = \{ t \mid t \in T_D^1 \vee t \in T_D^2 \}$, and

$T_E = \{ W \mid W \in T_D^1 \vee W \in T_D^2 \}$

The correctness of the union operator is established in the following theorem.

Theorem 6.4.5.1 $REPREP(REP(T_1 \cup T_2)) = REPREP(REP(T_1) \cup REP(T_2))$ for any consistent and reduced domain compatible E-tables T_1 and T_2.

Theorem 6.4.5.1 is demonstrated in Figure 6.26 through Figure 6.31.

6.4.6 Intersection

Definition 6.4.6.1: Let R be a relational scheme, let U_1 and U_2 be members of Σ_R, then $U_1 \cap U_2 = REDUCEREP(U)$, where U is obtained as follows:

$$U = \{ r \mid (\exists r_1)(\exists r_2)(r_1 \in U_1 \wedge r_2 \in U_2 \wedge r = \{ t \mid (t \in \lambda) \vee (t \notin \lambda \wedge t \in r_1 \wedge t \in r_2) \}) \}.$$

The intersection T of two E-tables T_1 and T_2 is obtained as follows: the common members of T_D^1 and T_D^2 constitute the members of T_D. For each set of tuple set W of T_E^1 (T_E^2), W is included in T_E if every non-dummy value of every tuple set of W is a member of T_D^2 (T_D^1). Formally, intersection is defined as a mapping $\cap : \Gamma_R \times \Gamma_R \rightarrow \Gamma_R$ as follows:

Definition 6.4.6.2 Let T_1 and T_2 be two consistent and reduced domain compatible E-tables, then

$T_1 \cap T_2 = REDUCE(CONTRADICTION(T))$, where

$T_D = \{ t \mid t \in T_D^1 \wedge t \in T_D^2 \}$, and

$$T_E = \{ W \mid (W \in T_E^1 \wedge (\forall w)(w \in W \rightarrow (\forall t)(t \in w \wedge t \notin \lambda \rightarrow t \in T_D^2))) \vee$$
$$(W \in T_E^2 \wedge (\forall w)(w \in W \rightarrow (\forall t)(t \in w \wedge t \notin \lambda \rightarrow t \in T_D^1))) \}$$

The correctness of the intersection operator is established in the following theorem.

Theorem 6.4.6.1 $REPREP(REP(T_1 \cap T_2)) = REPREP(REP(T_1) \cap REP(T_2))$ for any

consistent and reduced domain compatible E-tables T_1 and T_2.

Examples of the extended intersection operator and Theorem 6.4.6.1 are demonstrated in Figure 6.33, Figure 6.34, Figure 6.35, and Figure 6.36.

As a remark, in concluding this chapter, we note that for the regular relational intersection and difference operators, the following relationship holds for regular relations T_1 and T_2:

$T_1 \cap T_2 = T_1 - (T_1 - T_2)$

However, this relationship no longer holds under the E-table model with respect to the extended relational operators of intersection and difference defined in this chapter. This claim is supported by the following figure:

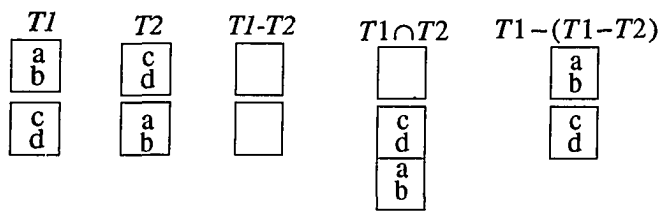


Figure 6.32: $T_1 \cap T_2 \neq T_1 - (T_1 - T_2)$

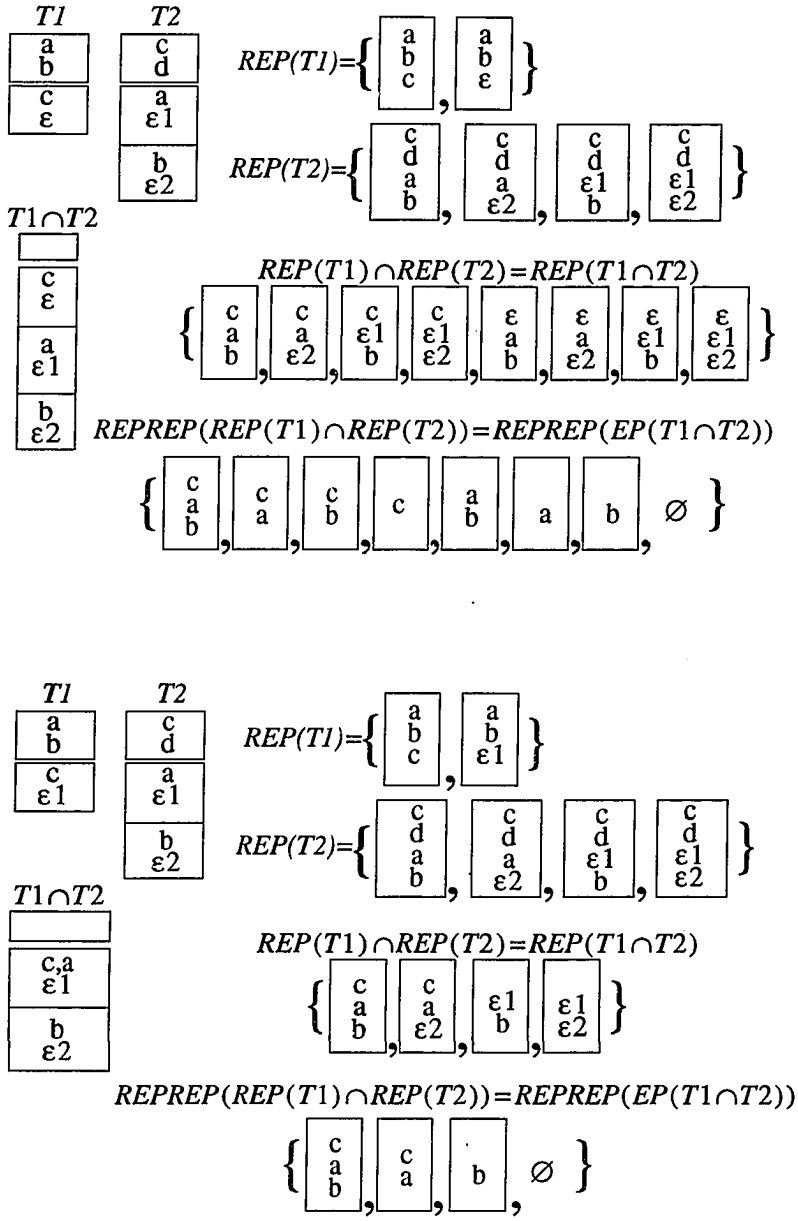
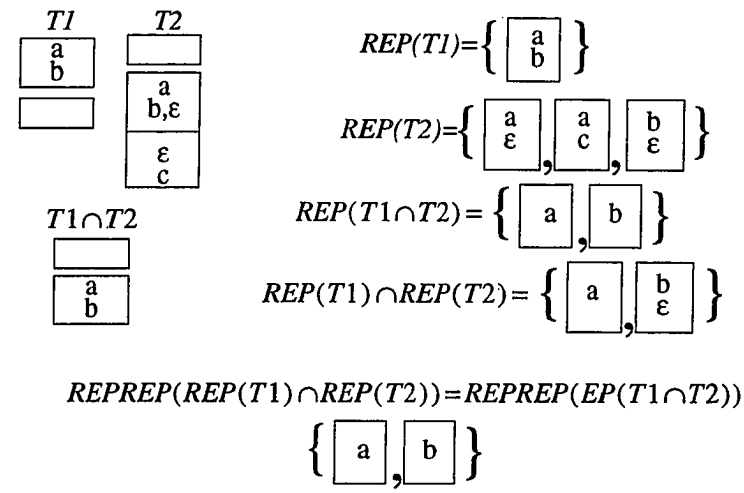


Figure 6.33: Intersection



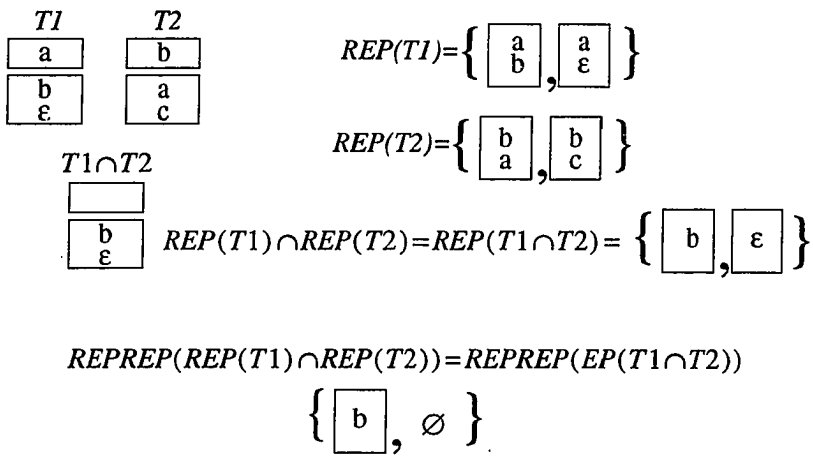


Figure 6.34: Intersection

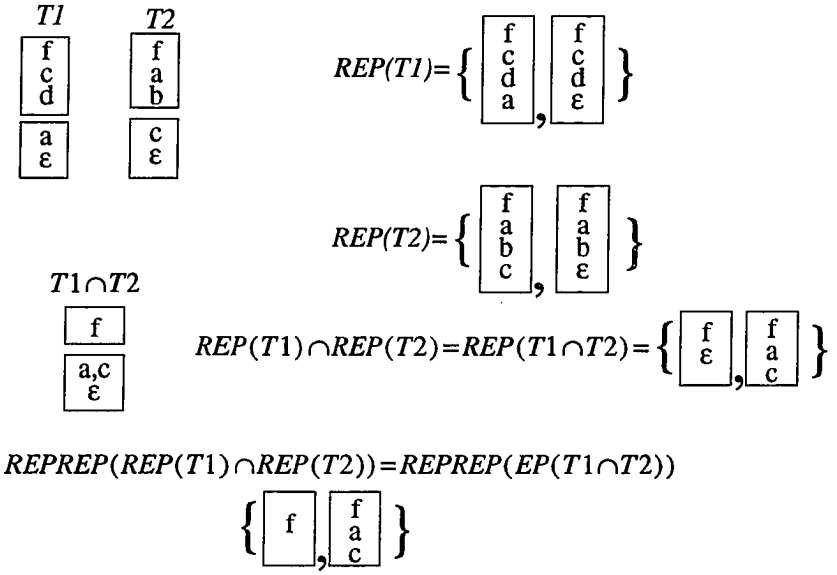


Figure 6.35: Intersection

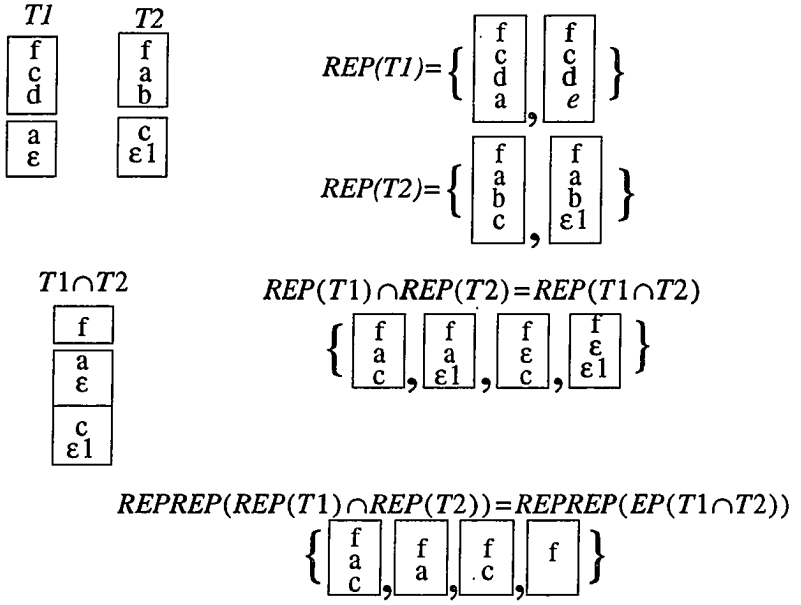


Figure 6.36: Intersection

7. CONCLUSION AND DISCUSSION

This dissertation addresses issues related to the recurrent and prominent problem of modeling null values in the form of inclusively and exclusively disjunctions within the realm of the relational model. Three extended relational models, namely I-tables, M-tables, and E-tables, for incorporating incomplete information are investigated and the problems raised by information incompleteness in the context of these three extended relational models, or how to meaningfully interpret and process incomplete information under the extended models, are explored

Chapters 3 through 5 extend previous works on the I-table and M-table models. Specifically, chapter three precisely defines the natural join relational operator on I-tables in a semantically correct manner. Furthermore, algorithms for computing the extended natural joins are examined with respect to efficiency, which is evaluated based upon the number of pair-up operations and I/O block accesses. As a result, an algorithm which requires a linear order of pair-up operations and a linear order of I/O activities is attained by using the "compare, concatenate, and then permute" strategy and by employing virtual relations with pointers.

In chapter 4, the update operations on I-tables are formalized and extended in a semantically correct manner. In order to retain the semantic correctness, it is necessary to extend the semantics, or information content, associated with I-tables. results obtained in this chapter indicate that the I-table model constitutes a representation system with respect to the update operations defined herein. For simplification, the insertion and deletion operations discussed allow only for inserting and deleting a single tuple and a single tuple set. However, the results obtained are readily extensible to more general forms of insertions and deletions. Note that the extension of relational operators with the extended semantics with the *OCCUR* function is quite straightforward. Also, the extended modification operation is not formalized since modifications

can be expressed as a sequence of deletions and insertions.

Efficient implementation of the extended update operations are vital to the feasibility of the I-table model. Though this issue is not discussed in chapter four, it can be shown that the extended insertion operations can be performed in logarithmic time and the extended deletion operations can be conducted in linear time.

Chapter 5 adds another dimension to the work described in chapter four. That is, the update operations of insertion and deletion are formalized and extended to M-tables, generalized I-tables, in a semantically correct manner.

A remark on the interpretation of the update operations in incomplete databases is perhaps appropriate at this point. We classify an update operation as an incomplete update if the operation is performed on incomplete data. For example, with respect to I-tables, an insertion is considered to be incomplete if a tuple is to be added into a tuple set of the indefinite component. The deletion operations are interpreted similarly. We feel that the mere appearance of the logical operator *OR* ($\vee$) does not always warrant the interpretation in the incompleteness sense.

In recent years, many different approaches have been proposed to extend the relational model to incorporate null values. One might raise the question that whether the I-table model is yet another such approach to null values that leads nowhere. The answers to this query is still open and have two facets. Firstly, I-tables model the indefiniteness in the form of disjunctive information and is different from approaches for null values. Secondly, we feel that the feasibility of the I-table model relies on the successful resolution of the following issues: (1) generalize relational operators to I-tables and correctly process queries issued against I-tables, (2) generalize update operations and correctly handle updates issued against I-tables, (3) generalize data dependencies to I-tables and correctly handle them within the context of I-tables, (4) generalize or identify normal forms within the context of I-tables, (5) generalize optimizations to the extended relational operators, and (6) implement an indefinite database

based on the I-table model. Issues 1 through 4 concerns with the theoretical feasibility of the I-table model while issues 5 and 6 ensures the empirical feasibility. Chapter three and chapter four has contributed towards the resolution of the first and the second issue.

The work presented in Chapter 6 is motivated by the goal to generalize the various null value approaches to represent the most general forms of incomplete information and the ultimate ambition to derive a uniformed relational approach for representing not only both inclusively and exclusively disjunctive information but also null values. The E-table model is intended to serve as a basis for this purpose. The E-table offers an extended relational approach for representing exclusively disjunctive information. The E-table approach models the disjunctive information on the tuple level thereof eliminates the limitation of the expressive power of most existing null value approaches. The relational algebra operators of selection, projection, cartisian product, difference, union, and and intersection are extended to E-tables. The extended relational operators are semantically correct in the Imielinski and Lipski ([ImLi84]) sense and are faithful in the sense that they reduce to the corresponding regular relational operators when E-tables contain only the definite components. The E-table approach precisely models null values of the type "value at present unknown, but one of some finite set of known possible values" discussed by Reiter [Reit84] in the form of exclusive disjunctions.

To digress for a moment: three major factors need to be considered when devising an extended relational model. These three factors are: (1) the ability of the extended relational model for supporting relational operators in a semantically correct manner (in the Imilieski and Lipski sense), (2) the simplicity of the notation and the definitions of the extended relational operators, and (3) the physical compactness in terms of data storage. We feel that the most important factor is whether an extended relational model is able to handle the relational operators in a semantically correct manner in the Imilieski and Lipski sense. The simplicity of the notation and definition is also an important factor for otherwise it would not have any

practical value. Physical storage consideration is secondary to semantic correctness and notational simplicity since the advancement of computing technology is capable of offering a vast amount of reliable disk storage and ever increasing main memory storage. Furthermore, CPU power is multiplying rapidly, especially with the parallel paradigm. We feel that the E-table model offers relatively simply notations. However, we do realize that the physical compactness aspect of the E-table model needs to be improved. Improvements can be attained at both the implementation level and modeling level. At the implementation level, techniques can be used to minimized physical storage inefficiency. At the modeling level, on the other hand, improvements can be achieved by further enhancements/modifications of the E-table model. One enhancement which would improve the physical storage is the incorporation of null values.

Returning to the topic of extended relational models per se: in related work, I-table models inclusively disjunctive information and is composed of three components: the definite component, the indefinite component, and the maybe component. The examples in Figure 7.1 submit comparisons of I-tables and E-tables. The I-table $T1$, whose definite and maybe components are both empty, represents three possible real world states ($POSS(REP(T1))$). However, the E-table $T2$ similar to $T1$ represents only two possible real world states ($REPREP(T2)$). Note that $\{a,b\}$ is not in $REPREP(T2)$ because of the exclusive semantics of the E-tables. Interestingly, the information content of the E-table $T3$ with special dummy values is superficially similar to that of the I-table $T1$. The difference lies in the fact that $\varnothing$ is also a possibility. Note that the information modeled by E-tables and I-tables is distinguishable while the two structures intersect when they contain only the definite component. E-tables are less complicated in the sense that it has one less component.

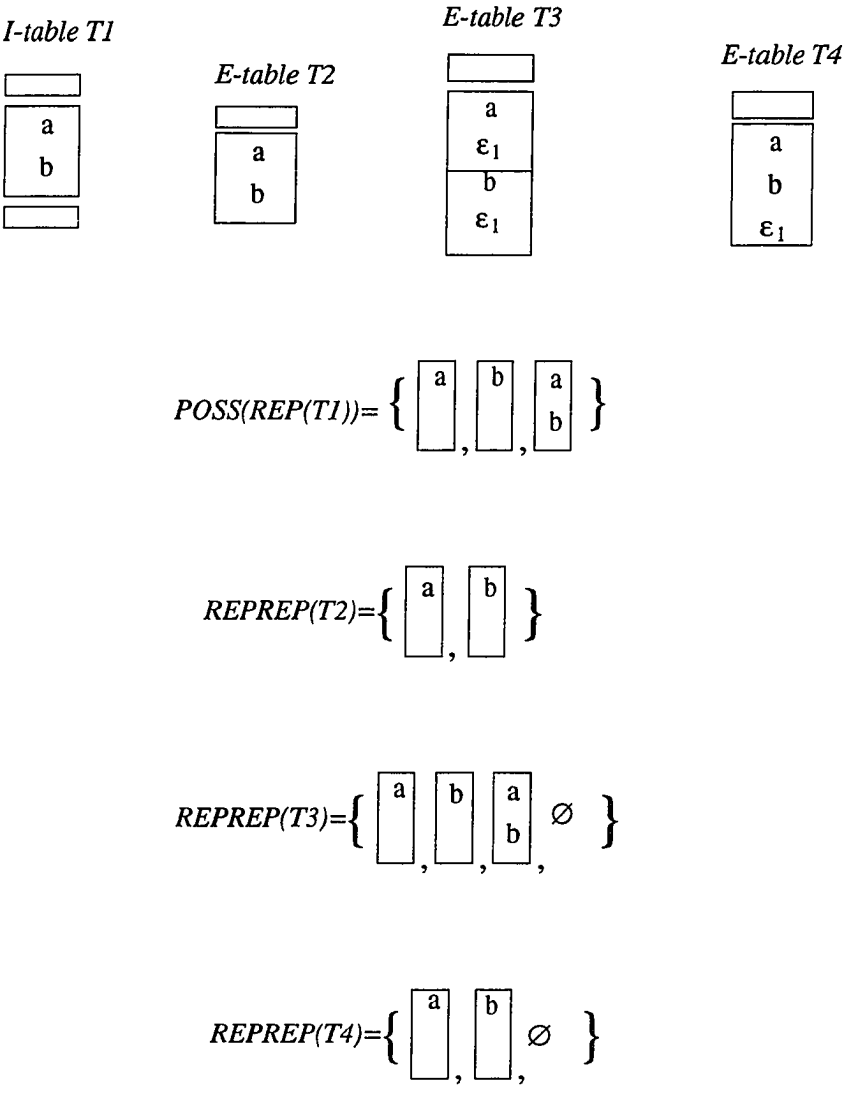


Figure 7.1: Comparison of E-tables and I-tables

Although E-tables do not have a maybe component, maybe information is suggested by the presence of dummy values in tuple sets. To provide a perspective on this aspect of dummy values, let us examine the E-table *T*4 in Figure 7.1. It can be seen, based on the information content of *T*4, both *a* and *b* are of the maybe flavor since the underlying information content

indicates that "nothing being true" is also a possibility. On the other hand, a and b in the E-table $T2$ do not belong to the category of maybe information and are indefinite. In other words, dummy values provide a different and more precise mechanism for representing maybe information. Another interesting aspect of dummy values is exhibited by the E-tables $T3$ and $T4$ in Figure 7.1. The possibility of both a and b being true is implied by $T3$ yet is excluded under $T4$. Therefore, in the world of positive information, $T3$ and $T4$ are unequal. However, the special case that "nothing being true" indicates that $\neg a$, $\neg b$, and $\neg(a \wedge b)$ are true under both $T3$ and $T4$ (while $\neg(a \wedge b)$ is implied by $T4$). Hence, in the world of negative information, $T3$ and $T4$ are implicitly equivalent.

One topic for future research is to generalize the E-table model to incorporate exclusively disjunctions of the form $P_1 \mid P_2 \mid \cdots \mid P_n$, where P_i's could be distinct predicates. It would also be interesting to explore the possibility of further extending the E-table model to append probability values for each tuple/tuple set. Note that such an extension would fold the definite and indefinite components of E-tables into just one component.

The problem of precisely characterizing what constitutes contradictions in the context of the generalized exclusive *or* is also a topic for further study. One direction for such effort is to represent E-tables in terms of graphs and then derive the necessary and sufficient conditions under which contradictions occur based on graphes. It should be mentioned in passing that an inconsistent E-table is reduced to empty sets as stated in chapter 6. This reduction is strictly based on the logic point of view. The alternatives are either removing the tuples/tuple sets that are in contradiction or removing one of the tuples/tuple sets in contradiction so that the resulting set of tuples/tuple sets is consistent.

In conclusion, the material presented on E-tables constitutes the first step towards our goal in attempting to capture both inclusively and exclusively disjunctive information in a uniformed relational approach and ultimately to derive a general model (uniformed) which is capable of

representing not only inclusive and exclusive disjunctions but also null values. One direction for continuing research therefore is to examine the possibility of incorporating the above mentioned incomplete information into a single model. Here, we provide a sketch of an extended relational model, IE-table (*I*nclusive and *E*xclusive table), for representing both inclusively and exclusively disjunctive information. It should be clear by now that I-tables model incomplete information in the form of inclusive disjunction, while E-tables model exclusively disjunctive type of incomplete information. Naturally, the logical question then is whether it is possible to represent both inclusively and exclusively disjunctive information under the relational approach. The research in this area presented in the literature ([MiGr88]) employs the set concept. [MiGr88]'s approach allows generalized sets (representing inclusively disjunctive information), disjunctive sets (representing exclusively disjunctive information), and collective sets (representing set values - that is every value inside a collective set is an actual value) as attribute values. The distinction of the corresponding type of each set can be tagged by a mapping from sets to values indicating the associated set types or can be stored in an extraneous tagging fields as a part of a relation itself. IE-table model offers a different viewpoint and adopts a uniformed relational approach without any extraneous tagging functions or attribute fields. The inspiration for IE-tables comes from the E-table model and the main idea is derived from the fact that, logically, inclusive disjunctions can be represented in terms of conjunctions of exclusive disjunctions. For instance, $a \vee b$ is equivalent to $a \bar{\vee} b \bar{\vee} (a \wedge b)$. Therefore, inclusive disjunctions can be explicitly represented as conjunctions of exclusive disjunctions, where each conjunction represents a combination of possibilities or potential real world truthes.

To provide a perspective on this concept, an E-table attempting to represent $a \vee b$ in terms of the generalized exclusive *or* is shown in Figure 7.2.

IE-table T

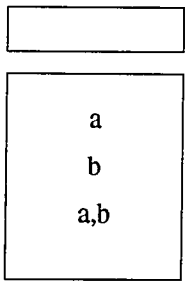


Figure 7.2: An Example of an IE-table

It can be inferred from Figure 7.2 that an IE-table is composed of two components: the definite component and the indefinite component. As always, the definite component represents definite information (e.g., the information which is known to be true). The indefinite component represents both inclusively and exclusively disjunctive information. The intended semantics for the IE-table illustrated in Figure 7.1 is that only one of the following is true: a, b, and $a \wedge b$. This semantic interpretation is the same as that of E-table's. However, according to Table 7.1, the logical formula $a \mid b \mid (a \wedge b)$ evaluates to false when both a and b are true. This certainly contradicts to the intended meaning.

Table 7.1: Truth Table for $a \mid b \mid (a \wedge b)$

a	b	$a \wedge b$	a	b	$(a \wedge b)$
0	0	0			0
0	1	0			1
1	0	0			1
1	1	1			u

To resolve this problem, we propose a concept called Tuple Set Closed World Assumption (TSCWA). Under TSCWA, a negative ground formula can be assumed to be true straightforwardly within the realm of a tuple set if it is in the domain of the very tuple set and its positive counterpart is not present in the very tuple set. In other words, positive facts, or atomic formulas, are explicitly stored in the tuple sets and negative facts are implicitly present provided that their corresponding positive counterpart cannot be proven from the positive facts explicitly stored in their corresponding tuple sets. Let us consider the example of Figure 7.2. The domain of the tuple set $\{\{a\},\{b\},\{a,b\}\}$ is $\{a,b\}$. Therefore, under TSCWA, the E-table of Figure 7.2 represents the following:

$(a \wedge \overline{b}) \mid (\overline{a} \wedge b) \mid (a \wedge b)$

Table 7.2: Truth table for the E-table of Figure 7.1 under TSCWA

a	b	$a \wedge b$	$a \wedge \overline{b}$	$\overline{a} \wedge b$	$(a \wedge \overline{b}) \mid (\overline{a} \wedge b) \mid (a \wedge b)$
0	0	0	0	0	0
0	1	0	0	1	1
1	0	0	1	0	1
1	1	1	0	0	1

Table 7.2 verifies that, under TSCWA, the E-table of Figure 7.1 is consistent with its intended meaning.

From now on, an IE-table is defined as an E-table together with TSCWA.

Figure 7.3 illustrates an IE-table representing $a \vee b \vee c$ which is validated by Table 7.3.

A more realistic example is exemplifies in Figure 7.4 and is validated by Table 7.4.

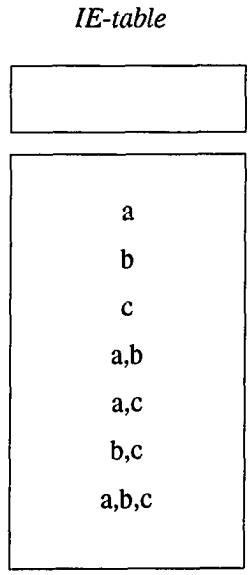


Figure 7.3: The IE-table representing $a \vee b \vee c$

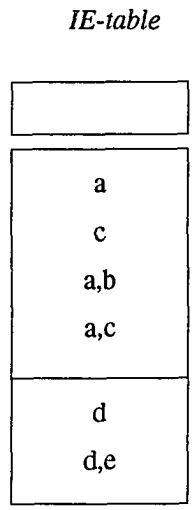


Figure 7.4: Another example of an IE-table

Table 7.3: Truth Table for $a \vee b \vee c$

a	b	c	$a \wedge \overline{b} \wedge \overline{c}$	$\overline{a} \wedge b \wedge \overline{c}$	$\overline{a} \wedge \overline{b} \wedge c$	$a \wedge b \wedge \overline{c}$	$a \wedge \overline{b} \wedge c$	$\overline{a} \wedge b \wedge c$	$a \wedge b \wedge c$
			A	B	C	D	E	F	G
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0	0	0
0	1	0	0	1	0	0	0	0	0
0	1	1	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	1	0	0
1	1	0	0	0	0	1	0	0	0
1	1	1	0	0	0	0	0	0	1

a	b	c	A	B	C	D	F	G	H
0	0	0				1			
0	0	1				1			
0	1	0				1			
0	1	1				1			
1	0	0				1			
1	0	1				1			
1	1	0				1			
1	1	1				1			

Table 7.4: Truth Table for the IE-table of Figure 7.4

a	b	c	d	e	$a \wedge \overline{b} \wedge \overline{c}$	$\overline{a} \wedge \overline{b} \wedge c$	$a \wedge b \wedge \overline{c}$	$a \wedge \overline{b} \wedge c$	$d \wedge \overline{e}$	$d \wedge e$
					A	B	C	D	E	F
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	1	0
0	0	0	1	1	0	0	0	0	0	1
0	0	1	0	0	0	1	0	0	0	0
0	0	1	0	1	0	1	0	0	0	0
0	0	1	1	0	0	1	0	0	1	0
0	0	1	1	1	0	1	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0
0	1	0	1	1	0	0	0	0	0	1
0	1	1	0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	1	0
0	1	1	1	1	0	0	0	0	0	1
1	0	0	0	0	1	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0
1	0	0	1	0	1	0	0	0	1	0
1	0	0	1	1	1	0	0	0	0	1
1	0	1	0	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	1	0
1	0	1	1	1	0	0	0	1	0	1
1	1	0	0	0	0	0	1	0	0	0
1	1	0	0	1	0	0	1	0	0	0
1	1	0	1	0	0	0	1	0	1	0
1	1	0	1	1	0	0	1	0	0	1
1	1	1	0	0	0	0	0	0	0	0
1	1	1	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	1	0
1	1	1	1	1	0	0	0	0	0	1

Table 7.4: Continued

a	b	c	d	e	$(A$	B	C	$D) \wedge (E$	$F)$
0	0	0	0	0				0	
0	0	0	0	1				0	
0	0	0	1	0				0	
0	0	0	1	1				0	
0	0	1	0	0				0	
0	0	1	0	1				0	
0	0	1	1	0				1	
0	0	1	1	1				1	
0	1	0	0	0				0	
0	1	0	0	1				0	
0	1	0	1	0				0	
0	1	0	1	1				0	
0	1	1	0	0				0	
0	1	1	0	1				0	
0	1	1	1	0				0	
0	1	1	1	1				0	
1	0	0	0	0				0	
1	0	0	0	1				0	
1	0	0	1	0				0	
1	0	0	1	1				0	
1	0	1	0	0				0	
1	0	1	0	1				0	
1	0	1	1	0				1	
1	0	1	1	1				1	
1	1	0	0	0				0	
1	1	0	0	1				0	
1	1	0	1	0				1	
1	1	0	1	1				1	
1	1	1	0	0				0	
1	1	1	0	1				0	
1	1	1	1	0				0	
1	1	1	1	1				0	

In summary, two lines of research can be contemplated from the above discussion. On the side of the modeling aspect, it is necessary to investigate the implication of the Tuple Set Closed World Assumption under the context of IE-tables and determine whether TSCWA is consistent with IE-tables. On the side of the relational aspect, it is necessary to extend relational operators to the IE-table model in a semantically correct manner.

REFERENCES

[AbGr85] Abiteboul S., and Grahne G. "Update Semantics for Incomplete Databases," *Proceedings of the 11th International Conference on Very Large Data Bases*, Stockholm, Sweden, 1985, pp. 1-12.

[AbKG87] Abiteboul S., Kanellakis P., and Grahne G. "On the Representation and Querying of Sets of Possible Worlds," *Proceedings of the ACM SIGMOD Conference*, 1987.

[Banc92] Bancilhon, F. "Understanding Object-Oriented Database Systems," *Proceedings of the Third International Conference on Extending Database Technology*, Vienna, Austria, March 1992, pp. 1-9.

[BaGP90] Barbara, D., Garcia-Molina, H., and Porter, D. "A Probabilistic Relational Data Model," in *Advances in Database Technology - International Conference on Extending Database Technology*, Venice, Italy, Bancilhon, F., Thanos, C., and Tsichritzis, D., Eds., Springer-Verlag, Berlin, March 26-30, 1990, pp. 60-74.

[BaKe88] Bakeland, R. and Kerre, E.E. "Piecewise Linear Fuzzy Quantities: A Way to Implement Fuzzy Information Into Expert Systems and Fuzzy Databases," in *Uncertainty and Intelligent Systems - 2nd International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Urbino, Italy, July 4-7, 1988, Bouchon, B., Saitta L., and Yager, R.R., Eds., Springer-Verlag, Berlin, 1988, pp. 120-126.

[Beec92] Beech, D. "A Foundation for Evolution from Relational to Object Databases," *Proceedings of the Third International Conference on Extending Database Technology*, Vienna, Austria, March 1992, pp. 251-270.

[Bisk81] Biskup J. "A Formal Approach to Null Values in Database Relations," in *Advances in Database Theory*, H. Gallaire, J. Minker, and J. M. Nicolas, Eds, Vol. 1, Plenum Press, New York and London, 1981, pp. 299-341.

[Bisk83] Biskup J. "Foundations of Codd's Relational Maybe Operations," *ACM Transactions on Database Systems*, Vol. 8, No. 4, 1983, pp. 608-636.

[BrCe91] Brodie, M.L. and Ceri, S., *Proccedings of the Second Intelligent and Cooperative Information Systems: Core Technology for Next Generation of Information Systems*, Como, Italy, October 1991.

[Brew90] Brewka, G., "Handling Incomplete Knowledge in Artificial Intelligence," *Proceedings of the First Workshop on Information Systems and Artificial Intelligence: Integration Aspects,* Ulm, France, March 19-21, 1990, pp. 11-29.

[BSAW91] Bonissone, P.P, Stillman, J.P., Aragone, J.K., and Wood, N.C. "Reasoning with Incomplete and Uncertain Information," *Technical Report*, General Electric Company, New York, 1991.

[BuP82a] Buckles, B.P. and Petry, F.E. "Fuzzy Databases and Their Applications," in *Fuzzy Information and Decision Processes*, M.M. Gupta and E. Sanchez, Eds, 1982, pp. 361-371.

[BuP82b] Buckles, B.P. and Petry, F.E. "A Fuzzy Representation of Data for Relational Databases," *Fuzzy Sets and Systems*, Vol. 7, 1982, pp. 213-226.

[BuP82c] Buckles, B.P. and Petry, F.E. "Extending the Fuzzy Database with Fuzzy Numbers," *Information Science*, Vol. 34, 1982, pp. 145-155.

[CaPi87] Cavallo, R. and Pittarelli, M. "The Theory of Probabilistic Databases," *Proceedings of the 13th International Conference on Very Large Data Bases*, Brighton, England, 1987, pp. 71-81.

[CaRo86] Cattell, R.G.G. and Rogers, T.R. "Combining Object-Oriented and Relational Models of Data," *Proceedings of the International Workshop on Object-Oriented Database Systems*, Pacific Grove, California, September 23-26, 1986, pp. 212-213.

[CaRS89] Cammarata, S., Ramachandra, P., and Shane, D. "Extending Relational Databases with Deferred Referential Integrity Checking and Intelligent Joins", *Proceedings of the 1989 ACM SIGMOD International Conference on the Management of Data*, Portland, Oregon, June, 1989, pp. 88-97.

[CDDG88] Charpentier, E., Daures, J.P., Dujols, P., and Gremy, F. "An Intuitive Representation of Imperfect Information," in *Uncertainty and Intelligent Systems - 2nd International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Urbino, Italy, July 4-7, 1988, Bouchon, B., Saitta L., and Yager, R.R., Eds., Springer-Verlag, Berlin, 1988, pp. 322-329.

[ChFM87] Chakravarthy, U.S., Fishman D.H., and Minker, J. "Semantic Query Optimization in Expert Systems and Database Systems," in *Expert Database Systems*, Larry Kerschberg, Ed., Benjamin/Cummings, Menlo Park, 1987, pp. 659-674.

[ChMG87] Chakravarthy, U.S., Minker, J. and Grant, J. "Semantic Query Optimization: Additional Constraints and Control Strategies," in *Expert Database Systems*, Larry Kerschberg, Ed., Benjamin/Cummings, Menlo Park, 1987, pp. 345-379.

[Chol88] Cholvy L., "A Model Approach To Update Semantic Problem," in *Data and Knowledge*, R.A. Meersman, and A.C. Sernadas, Eds, North-Holland, Amsterdam, 1988, pp. 89-98.

[Clar78] Clark, K. "Negation as Failure," in *Logic and Databases*, H. Gallaire and J. Minker, Eds., Plenum, New York, 1978, pp. 293-322.

[Codd70] Codd E.F. "A Relational Model for Large Shared Data Banks," *Communications of the ACM*, Vol. 13, No. 6, June 1970, pp. 377-387.

[Codd75] Codd, E.F. "Understanding Relations," *FDT Bulletin of ACM-SIGMOD*, Vol. 7, 1975, pp. 23-28.

[Codd79] Codd E.F. "Extending the Database Relational Model to Capture More Meaning," *ACM Transactions on Database Systems*, Vol. 4, No. 4, 1979, pp. 397-434.

[Comm90] The Committee for Advanced DBMS Function "Third Generation Data Base System Manifesto," *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, Atlantic City, NJ, May 23-25, 1990, pg. 396.

[CuHu91] Cussens, J. and Hunter, A. "Using Defeasible Logic for a Window on a Probabilistic Database: Some Preliminary Notes," in *Symbolic and Quantitative Approaches to Uncertainty - Proceedings of the European Conference ECSQAU*, Marseille, France, October 15-17, 1991, pp. 146-152.

[Ditt86] Dittrich, K.R. "Object-Oriented Database Systems: the Notion and the Issues," *Proceedings of the International Workshop on Object-Oriented Database Systems*, Pacific Grove, California, September 23-26, 1986, pp. 2-4.

[FaUV83] Fagin F., Ullman. J.D., and Vardi M.Y., "On the Semantics of Updates in Databases," *Proceedings of ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, Atlanta, 1983, pp. 352-365.

[GaMN84] Gallaire, H., Minker, J., and Nicolas, J.M. "Logic and Databases: A Deductive Approach," *ACM Computing Survey*, Vol. 16, No. 2, June 1984, pp. 151-184.

[GaNP92] Gadia, S.K., Nair, S.S., and Poon, Y.C. "Incomplete Information in Relational Temporal Databases," *Proceedings of the 18th International Conference on Very Large Data Bases*, Vancouver, British Columbia, Canada, 1992, pp. 395-406.

[GeBP91] George, R., Buckles, B.P., and Petry, F.E. "An Object-Oriented Data Model To Represent Uncertainty in Coupled Artificial Intelligence - Database Systems," in *The Next Generation of Information Systems: From Data to Knowledge (A Selection of Papers Presented at Two IJCAI-91 Workshops, Sydney, Australia, August 1991)*, Papazoglou, M.P. and Zeleznikow, J., Eds., pp. 37-48.

[GeHe86] Gelenbe, E. and Hebrail, G. "A Probability Model of Uncertainty in Data Bases," *Proceedings of the International Conference on Data Engineering*, February 5-7, 1986, Los Angeles, California, 1986, pp. 328-333.

[GoNg85] Goodman, I.R. and Nguyen, H.T., *Uncertainty Models for Knowledge-Based System*, North-Holland, Amsterdam and New York, 1985.

[Gotl75] Gotlieb, L.R. "Computing Joins of Relations," *ACM SIGMOD International Conference on Management of Data*, pp. 55-63, 1975.

[Gran79] Grant, J. "Partial Values in a Tabular Database Model," *Information Processing Letter*, Vol. 9, No. 2, August 1979, pp. 97-99.

[Grah91] Grahne, G. "The problem of Incomplete Information in Relational Databases," in *Lecture Notes in Computer Science*, No. 554, Springer-Verlag, Berlin, 1991.

[GrMi92] Grant, J. and Minker, J. "The Impact of Logic Programming on Databases," *Communications of the ACM*, Vol. 3, No. 3, 1992, pp. 67-81.

[GuMa87] Gupta, A. and Madnick, S. "Object-Oriented Approach to Integrating Database Semantics," *Technical Report*, MIT, Cambridge, 1987.

[HePa87] Henschen, L.J. and Park, H. "Compiling the GCWA in Indefinite Deductive Databases," in *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed., Morgan Kaufmann, Los Altos, 1987, pp. 395-438.

[Heue88] Heuer, A. "Foundations of Relational Object Management Systems," in *Advances in Object-Oriented Database Systems - Proceedings of the Second International*

Workshop on Object-Oriented Database Systems, Bad Munster am Stein-Ebernburg, FRG, September 27-30, 1988, pp. 209-212.

[HeYa85] Henschen, L.J. and Yahya, A. "Deduction in Non-Horn Databases," *Journal of Automated Reasoning*, Vol. 1, 1985, pp. 395-438.

[Hill86] Hillyer, B.K. "Computationally Inexpensive Hash Functions for Relational Join," *Technical Memorandum*, AT&T Bell Laboratories, Homdel, October 26, 1986.

[HMMS92] Higa, K., Morrision, M., Morrison, J., and Sheng, O.R. "Object-Oriented Methodology for Knowledge Base/Database Coupling," *Communications of the ACM*, Vol. 35, No. 6, June 1992, pp. 99-113.

[HoWo89] Hong, W. and Wong, E. "Multiple Query Optimization Through State Transition and Decomposition," *Memorandum*, Electronics Research Laboratories, University of California, Berkeley, March 1989.

[HuKi87] Hull, R. and King, R. "semantic Database Modeling: Survey, Applications, and Research Issues," *ACM Computing Surveys*, Vol. 19, No. 3, September 1987, pp. 201-260.

[Imie89] Imielinski T. "Incomplete Information in Logical Databases," *Bulletin of the IEEE Tech. Committee on Data Engineering*, Vol. 12, No. 2, 1989, pp. 29-40.

[ImLi84] Imielinski T., and Lipski W. "Incomplete Information in Relational Databases," *Journal of the ACM*, Vol. 31, No. 4, October 1984, pp. 761-791.

[IoKa90] Ioannidis, Y.E. and Kang Y.C. "Randomized Algorithms for Optimizing Large Join Queries," *Proceedings of the 1990 ACM SIGMOD Internation Conference on Management of Data*, May 23-25, 1990, Atlantic City, New Jersey, pp. 312-321.

[JaKo84] Jarke, M. and Koch J. "Query Optimization in Database Systems," *Computing Surveys*, Vol. 16, No. 2, June 1984, pp. 111-152.

[Jark87] Jarke, M. "External Semantic Query Simplification: A Graph-Theoretic Approach and Its Implementation in Prolog," in *Expert Database Systems*, Larry Kerschberg, Ed, Benjamin/Cummings, Menlo Park, 1987, pp. 675-692.

[Kers90] Kerschberg, L., "Expert Database Systems: Knowledge/Data Management Environments for Intelligent Information Systems," *Information Systems*, Vol. 15, No. 1, pp. 151-160, 1990.

[Kim80a] Kim, W. "A New Way to Compute the Product and Join of Relations," *Proceedings of the ACM-SIGMOD International Conference on Management of Data*, Santa Monica, CA., May 14-16, ACM, New York, pp. 179-187.

[Kim80b] Kim, W. "Query Optimization for Relational Database Systems," *Report*, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana-Champaign, October 1980.

[Kimw90] Kim, W. "Research Directions in Object-Oriented Database Systems," *Proceedings of the Ninth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Nashville, Tennessee, April 2-4, 1990, pp. 1-15.

[King81] King, J.J. "QUIST: A System for Semantic Query Optimization in Relational Databases," *Proceedings of the 7th International Conference on Very Large Data Bases*, IEEE, New York, pp. 510-517.

[Kowa78] KowaLski, R. "Logic for Data Description," in *Logic and Databases*, H. Gallaire and J. Minker, Eds. Plenum, New York, 1978, pp. 77-103.

[KuKu91] Kulenovic, A.L. and Kulenovic, A. "Treatment of Null Values in the NFSet Data Model (A Relational Model Based on Sets and Lists)," in *Aspects of Databases - the Proceedings of the Ninth British National Conference on Databases*, 1991, Jackson, M.S. and Robinson, A.E., Eds. 1991, pp. 226-244.

[LaLy89] Lamport, L. and Lynch, N. "Chapter on Distributed Computing," *Technical Report*, MIT Laboratory for Computer Science, Cambridge, February 1989.

[Leve84] Levesque H.J. "The Logic of Incomplete Knowledge Bases," in *On Conceptual Modeling*, Brodie, M. Mylopoulos, J., and Schmidt, J., Eds, Springer-Verlag, Berlin, 1984, pp. 165-186.

[Lips79] Lipski W. "On Semantic Issues Connected with Incomplete Information," *ACM Transaction on Database Systems*, Vol. 4, No. 3, September 1979, pp. 262-296.

[LiSu88] Liu K.C., and Sunderraman R. "On Representing Indefinite and Maybe Information in Relational Databases," *Proceedings of the Fourth International Conference on Data Engineering*, Los Angeles, CA, February 2-4, 1988, pp. 250-257.

[LiS90a] Liu, K.C. and Sunderraman, R. "On Representing Indefinite and Maybe Information in Relational Databases: A Generalization," *Proceedings of the sixth International Conference on Data Engineering*, Los Angeles, CA, February 5-9, 1990, pp. 495-502.

[LiS90b] Liu K.C., and Sunderraman R. "Indefinite and Maybe Information in Relational Databases," *ACM Transactions on Database Systems*, Vol. 15, No. 1, March 1990, pp. 1-39.

[LiSu91] Liu, K.C. and Sunderraman, R. "A Generalized Relational Model for Indefinite and Maybe Information," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 3, No. 1, March 1991.

[LiZh91] Liu K.C., and Zhang L. "Natural Joins in Relational Databases with Indefinite and Maybe Information", *Proceedings of the Seventh International Conference on Data Engineering*, Kobe, Japan, April 8-12, 1991, pp. 132-139.

[LiZh92] Liu K.C., and Zhang L. "Updates in Relational Databases with Indefinite and Maybe Information", *Proceedings of the Second International Computer Science Conference - Data and Knowledge Engineering: Theory and Applications*, Hong Kong, Dec. 13-16, 1992, pp. 509-515.

[Maie83] Maier, D. *The Theory of Relational Databases*, Computer Science Press, Rockville, 1983.

[Maie86] Maier, D. "Why Object-Oriented Databases Can Succeed Where Others Have Failed," *Proceedings of the International Workshop on Object-Oriented Database Systems*, Pacific Grove, California, September 23-26, 1986, pg. 227.

[MaWa87] Manchanda, S. and Warren, D.S. "A Logic-based Language for Database Updates", in *Foundations of Deductive Databases and Logic Programming*, Minker, J., Ed., Morgan Kaufmann, Los Altos, 1987, pp. 363-394.

[MaZd87] Malley, C.V. and Zdonik, S.B. "A Knowledge-Based Approach to Query Optimization," in *Expert Database Systems*, Larry Kerschberg, Ed, Benjamin/Cummings, Menlo Park, 1987, pp. 329-343.

[MiGr88] Michalewicz, Z. and Groves, L.J. "Sets and Uncertainty in Relational Databases," in *Lecture Notes in Computer Science*, 313, Goos, G. and Hartmanis, J., Eds, Spring-Verlag, Berlin, 1988, pp. 127-137.

[Mink82] Minker, J. "On Indefinite Databases and the Closed World Assumption," *Proceedings of the Sixth Conference on CADE*, D. Loveland, Ed., Lecture Notes in Computer Science, No. 138, Springer-Verlag, Berlin, 1982, pp. 292-308.

[MiPe84] Minker, J. and Perlis, D. "Applications of Protected Circumscription," *Proceedings of the Conference of Automated Deduction*, Springer-Verlag, Berlin, 1984, pp. 414-425.

[MiWi86] Missikoff, M. and Wiederhold, G. "Towards a Unified Approach for Expert and Database Systems," in *Expert Database Systems*, Kerschberg, L. Ed., Benjamin/Cummings, Menlo Park, 1987, pp. 383-399.

[MoCa90] Moral, S. and de Campos, L.M. "Updating Uncertain Information," in *Uncertainty in Knowledge Bases - Proceedings of the 3rd International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, IPMU'90, Paris, France, July 1990, pp. 58-67.

[MoSi88] Mohany, H. and Siklossy, L. "Incomplete Databases and Some Applications," *Report*, Vrije Universiteit Amsterdam, Amsterdam, October, 1988.

[MyBr90] Mylopoulos, J. and Brodie, M., "Knowledge Bases and Databases: Current Trends and Future Directions," *Proceedings of the First Workshop on Information Systems and Artificial Intelligence: Integration Aspects*, Ulm, France, March 19-21, 1990, pp. 153-180.

[PaZe91] Papazoglou, M.P. and J. Zeleznikow, J. "The Next Generation of Information Systems - From Intelligence to Distribution and Cooperation," in *The Next Generation of Information Systems: From Data to Knowledge (A Selection of Papers Presented at Two IJCAI-91 Workshops, Sydney, Australia, August 1991)*, Papazoglou, M.P. and Zeleznikow, J., Eds., pp. 1-8.

[PeRR91] Perrizo, W., Rajkumar, J., and Ram, P. "Hydro: A Heterogeneous Distributed Database System," *Proceedings of the 1991 ACM SIGMOD International Conference on Management of Data*, Denver, Colorado, May 29-31, 1991, pp. 32-39.

[Przy87] Przymusinski, T.C. "On the Declarative Semantics of Deductive Databases and Logic Programs," in *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed., Morgan Kaufmann, Los Altos, 1987, pp. 193-216.

[QuMa91] Quinio, P. and Matsuyama, T. "Random Closed Sets: a Unified Approach to the Representation of Imprecision and Uncertainty," in *Symbolic and Quantitative Approaches to Uncertainty - Proceedings of the European Conference ECSQAU*, Marseille, France, October 15-17, 1991, pp. 282-286.

[RaPr87] Rakesh, A. and Premkumar D. "Moving Selections into Linear Fixpoint Queries," *Technical Memorandum*, AT&T Bell Laboratories, Homdel, July 15, 1987.

[ReFS92] Read, R.L., Fussel, D.S., and Silberschatz, A. "A Multi-Resolution Relational Data Model," *Proceedings of the 18th International Conference on Very Large Data Bases*, Vancouver, British Columbia, Canada, 1992, pp. 139-150.

[Reit78] Reiter, R. "On Closed World Data Bases," in *Logic and Databases*, H. Gallaire and J. Minker, Eds. Plenum, New York, 1978, pp. 56-76.

[Reit84] Reiter, R. "Towards a Logical Reconstruction of Relational Database Theory," in *On Conceptual Modeling*, Brodie, M. Mylopoulos, J., and Schmidt, J., Eds, Springer-Verlag, Berlin, 1984, pp. 191-233.

[Reit88] Reiter, R. "On Formalizing Database Updates: Preliminary Report," in *Advances in Database Technology - Proceedings of the International Conference on Extending Database Technology*, Venice, Italy, March 1988, pp. 10-20.

[Seli86] Selis, T.K. "Global Query Optimization," *Memorandum*, Electronics Research Laboratory, University of California, Berkeley, March 1986.

[Shep87] Shepherdon, J.H. "Negation in Logic Programming," in *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed., Morgan Kaufmann, Los Altos, 1987, pp. 19-88.

[ShMe90] Shenoi, S. and Melton, A. "An Extended Version of the Fuzzy Relational Database Model," *Information Sciences*, Vol. 52, 1990, pp. 35-52.

[Smit86] Smith, J.M. "Expert Database Systems: A Database Perspective," in *Expert Database Systems*, Kerschberg, L. Ed., Benjamin/Cummings, Menlo Park, 1987, pp. 3-15.

[Swam89] Swami, A. "Optimization of Large Join Queries: Combining Heuristics and Combinatorial Techniques," *Proceedings of the 1989 ACM SIGMOD International Conference on the Management of Data*, Portland, Oregon, June, 1989, pp. 367-376.

[ThRN87] Thom, J.A., Ramamohanarao, K., and Naish, L. "A Superjoin Algorithm for Deductive Databases", in *Foundations of Deductive Databases and Logic Programming*, Minker, J., Ed., Morgan Kaufmann, Los Altos, 1987, pp. 519-543.

[Ullm82] Ullman, J.D., *Principles of Database Systems*, Computer Science Press, Rockville, 1982.

[Ullm84] Ullman J.D. "Implementation of Logical Query Languages for Databases," *Technical Report*, Department of Computer Science, Stanford University, May 1984.

[Vade91] Vandenberghe, R.M. and de Caluwe, R.M. "An Entity-Relationship Approach to the Modeling of Vagueness in Databases," in *Symbolic and Quantitative Approaches to Uncertainty - Proceedings of the European Conference ECSQAU*, Marseille, France, October 15-17, 1991, pp. 338-343.

[VanG87] Van Gelder, A. "Negation as Failure Using Tight Derivations for General Logic Programming," in *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed., Morgan Kaufmann, Los Altos, 1987, pp. 149-176.

[Vass79] Vassiliou Y. "Null Values in Database Management: A Denotational Semantics Approach," *Proceedings of ACM-SIGMOD International Conference on the Management of Data*, Boston, May-June, 1979, pp. 162-169.

[Win86a] Winslett M.S., "A Model-Theoretic Approach to Updating Logical Databases," *Proceedings of the Fifth ACM PODS*, Cambridge, March, 1986, pp. 224-234.

[Win86b] Winslett M.S., "Is Belief Revision Harder Then You Thought?" *Proceedings of the 5th National Conference on Artificial Intelligence*, Philadelphia, August 1986, pp. 421-427.

[Win86c] Winslett M.S., "Updating Logical Databases Containing Null Values," *Proceedings of International Conference on Database Theory*, Rome, Italy, September, 1986, pp. 421-435.

[Wins90] Winslett M.S., *Updating Logical Databases*, Cambridge University Press, Cambridge, 1990.

[YaHe85] Yahya, A. and Henschen, L.J. "Deduction in Non-Horn Databases," *Journal of Automated Reasoning*, Vol. 1, 1985, pp. 141-160.

[YaLe91] Yang, J.D. and Lee, Y.J., "Characterization of Unknown Values with Implicit Predicate," *Decision Support System*, Vol. 7, 1991, pp. 133-144.

[Yaos79] Yao, S.B. "Optimization of Query Evaluation Algorithms," *ACM Transactions on Database Systems*, Vol. 4, No. 2, June 1979, pp. 133-155.

[Zani84] Zaniolo C. "Database Systems with Null Values," *Journal of Computer and System Sciences*, Vol. 28, 1984, pp. 142-166.

[Zede65] Zedeh, L.A. "Fuzzy Sets," *Information and Control*, Vol. 8, 1965, pp. 338-353.

[ZhLi92] Zhang L. and Liu, K.C., "Updates in Generalized Relational Databases with Indefinite and Maybe Information", *Proceedings of the International Computer Symposium*, Taichung, Taiwan, Dec. 12-15, 1992, pp. 790-797.

[ZhLi93] Zhang L. and Liu, K.C. "Towards Relational Model for Exclusively Disjunctive Information", *Proceedings of 1993 ACM Computer Science Conference*, Indianapolis, Indiana, Feb. 16-18, 1993, pp. 143-150.

[ZvCh86] Zvieli, A. and Chen P.P. "Entity - Relationship Modeling and Fuzzy Databases," *Proceedings of the International Conference on Data Engineering*, February 5-7, 1986, Los Angeles, California, 1986, pp. 320-327.

ACKNOWLEDGEMENTS

First of all, I would like to take this opportunity to thank and praise The Lord who stood by me, encouraged me, and helped me to complete this dissertation. Without His grace, this endeavor would be impossible. This dissertation is a testimony of the strength, the wisdom, and the faith with that He has blessed me.

My deepest appreciation is due to Dr. Ken-Chih Liu for the guidance he has provided throughout my academic career. The discussions we had have enriched my understanding of logic and databases. I also feel immeasurable gratitude for Dr. Liu, whose personal sacrifice has made it possible for me to continue my research and most importantly achieve my educational goal.

My heartfelt appreciation is due to Dr. Les Miller for his sound advice (both academic and administrative), scholarly comments and suggestions, and careful readings. I am also grateful to him, among many other things, for serving as my major professor. This has made it possible for me to finish my graduate study at Iowa State University and achieve my educational goal.

My special thanks are also due to Ms. Trish Stauble for carrying out the administrative details for which I would have been responsible if I were physically close to campus.

I would also like to put on record my intellectual debt to those researchers from whose work I have been benefited. Although it would not be feasible to attribute to each of them, the reference hopefully will furnish this purpose.

I must thank my parents for giving me the opportunity for a higher education and for their constant support and encouragement without which I would have found it difficult to achieve my goals.

Finally, I offer my warmest and heartiest thanks to my wife for, among many other things, her encouragement, understanding, and patience throughout the long writing period. This is hers

as much as mine.

APPENDIX

Theorem 3.2.1: Let R_1 and R_2 be two relational schemes with some common attributes. Then $<U_1,v_1> \infty <U_2,v_2> = REDUCEREP(<U_1,v_1>) \infty REDUCEREP(<U_2,v_2>)$ for any $<U_1,v_1> \in \Sigma_{R_1}$ and $<U_2,v_2> \in \Sigma_{R_2}$.

Proof:

Let $<U_1,v_1>$ be arbitrary element of Σ_{R_1} and $<U_2,v_2>$ be arbitrary element of Σ_{R_2}.

Let $<U_1,v_1> \infty <U_2,v_2> = <U,v>$, and

$REDUCEREP(<U_1,v_1>) \infty REDUCEREP(<U_2,v_2>) = <U',v'>$.

Let $U_1 = U_\alpha^1 \cup U_\beta^1$ such that $U_\alpha^1 \cap U_\beta^1 = \varnothing$ and

(1) $(\forall r_1)(r_1 \in U_\beta^1 \rightarrow (\exists r_2)(r_2 \in U_\alpha^1 \wedge r_2 \subset r_1))$,

(2) $(\forall r_1)(r_1 \in U_\alpha^1 \rightarrow \neg(\exists r_2)(r_2 \in U_\alpha^1 \wedge r_2 \subset r_1))$.

Similarly, let $U_2 = U_\alpha^2 \cup U_\beta^2$ such that $U_\alpha^2 \cap U_\beta^2 = \varnothing$ and

(3) $(\forall r_1)(r_1 \in U_\beta^2 \rightarrow (\exists r_2)(r_2 \in U_\alpha^2 \wedge r_2 \subset r_1))$,

(4) $(\forall r_1)(r_1 \in U_\alpha^2 \rightarrow \neg(\exists r_2)(r_2 \in U_\alpha^2 \wedge r_2 \subset r_1))$.

First we prove that $U = U'$:

Let r_j be an arbitrary element of U_β^1. By (1) there is a r_i in U_α^1 such that $r_i \subset r_j$. Hence, by the definition of the extended natural join on Σ, $r_i \infty r_k \subset r_j \infty r_k$, for any $r_k \in U_2$.

Similarly, for any r_j of U_β^2, by (3) there exists a r_i in U_α^2 such that $r_i \subset r_j$, and $r_i \infty r_k \subset r_j \infty r_k$, for any $r_k \in U_1$.

Therefore, by the definition of the extended natural join on Σ and the definition of *REDUCEREP*, $U = U'$.

Next we prove that $v = v'$:

First we show the following lemma.

Lemma 1: Let $<U,v> \in \Sigma_R$ and $REDUCEREP(<U,v>) = <U_{1,v_1}>$, then $\bigcup_{r\in U}(r)\cup v = \bigcup_{r\in U_1}(r) \cup v_1$.

Proof:

Let $U = U_\alpha \cup U_\beta$ such that $U_\alpha \cap U_\beta = \varnothing$ and

(1) $(\forall r_1)(r_1 \in U_\beta^1 \rightarrow (\exists r_2)(r_2 \in U_\alpha^1 \wedge r_2 \subset r_1))$,

(2) $(\forall r_1)(r_1 \in U_\alpha^1 \rightarrow \neg(\exists r_2)(r_2 \in U_\alpha^1 \wedge r_2 \subset r_1))$.

Let $U_\gamma = \{t \mid (\exists r_1)(\exists r_2)(r_1 \in U_\alpha \wedge r_2 \in U_\beta \wedge r_1 \subset r_2 \wedge t \in (r_2 - r_1))\}$

Then, $\bigcup_{r\in U}(r) \cup v = \bigcup_{r\in U_\alpha}(r) \cup \bigcup_{r\in \beta}(r) \cup v$

$= \bigcup_{r\in U_\alpha}(r) \cup \bigcup_{r\in U_\gamma}(r) \cup v$.

By the definition of *REDUCEREP*, $U_1 = U_\alpha$ and $v_1 = v \cup U_\gamma$.

Therefore, $\bigcup_{r\in U}(r)\cup v = \bigcup_{r\in U_1}(r)\cup v_1$. □

Thus, by Lemma 1 and the definition of the extended natural join on Σ, $v = v'$. □

Theorem 3.2.2: Let T_1 and T_2 be two I-tables under relational schemes R_1 and R_2 such that $T_I^1 = \{w_1^1,\ldots,w_m^1\}$ and $T_I^2 = \{w_1^2,\ldots,w_n^2\}$. Then $REP(T_1 \infty T_2) = REP(T_1) \infty REP(T_2)$.

Proof:

By the definition of *REP* and Theorem 3.2.1, we need to prove:

$REDUCEREP(<MM(T_1 \infty T_2), M(T_1 \infty T_2)>) =$

$<MM(T_1), M(T_1)> \infty <MM(T_2), M(T_2)>$

Let $REDUCEREP(<MM(T_1 \infty T_2), M(T_1 \infty T_2)>) = <U,v>$, and

$<MM(T_1), M(T_1)> \infty <MM(T_2), M(T_2)> = REDUCEREP(<U^o, v^o> = <U', v'>$.

First we prove that $U = U'$:

Let $U_1 = U_\alpha^1 \cup U_\beta^1$ such that $U_\alpha^1 \cap U_\beta^1 = \varnothing$ and

(1) $(\forall w_1)(w_1 \in U^1_\beta \rightarrow (\exists w_2)(w_2 \in U^1_\alpha \wedge w_2 \subset w_1))$,

(2) $(\forall w_1)(w_1 \in U^1_\alpha \rightarrow \neg(\exists w_2)(w_2 \in U^1_\alpha \wedge w_2 \subset w_1))$.

Similarly, let $U_2 = U^2_\alpha \cup U^2_\beta$ such that $U^2_\alpha \cap U^2_\beta = \varnothing$ and

(3) $(\forall w_1)(w_1 \in U^2_\beta \rightarrow (\exists w_2)(w_2 \in U^2_\alpha \wedge w_2 \subset w_1))$,

(4) $(\forall w_1)(w_1 \in U^2_\alpha \rightarrow \neg(\exists w_2)(w_2 \in U^2_\alpha \wedge w_2 \subset w_1))$.

Let w_j be an arbitrary element of U^1_β. By (1) there is a w_i in U^1_α such that $w_i \subset w_j$. Hence, by the definition of the extended natural join on Γ, $w_i \infty w_k \subset w_j \infty w_k$, for any $w_k \in U_2$. Similarly, for any w_j of U^2_β, by (3) there exists a w_i in U^2_α such that $w_i \subset w_j$, and $w_i \infty w_k \subset w_j \infty w_k$, for any $r_k \in U_1$.

Therefore, by the definition of the extended natural join on Γ and the definition of *REDUCEREP*, $U = U'$.

Next we prove that $v = v'$:

(1) Let t be an arbitrary element of $M(<T_1 \infty T_2>)$. Then there must exist a t_1 in T_1 and a t_2 in T_2 such that $t = t_1 \infty t_2$.

Case 1: $t_1 \in T^1_D$ and $t_2 \in T^2_M$. Then, there exists a r in $MM(T_1)$ such that $t_1 \in r$ and $t_2 \in M(T_2)$. Hence, by the definition of ∞ on Σ, $t \in v^o$.

Case 2: $t_1 \in T^1_M$ and $t_2 \in T^2_D$. Similar to Case 1, $t \in v^o$.

Case 3: $t_1 \in T^1_M$ and $t_2 \in T^2_M$. This implies that $t_1 \in M(T_1)$ and $t_2 \in M(T_2)$. Hence, by the definition of ∞ on Σ, $t \in v^o$.

Case 4: $(\exists w)(w \in T^1_I)$ and $w = \{s_{1,\ldots,} s_k\}$ and $t_1 \in w$ and $t_2 \in T^2_M$. This implies that $w \in MM(<T_{1>})$ and therefore $t_1 \in MM(<T_1>)$. Hence, $t \in v^o$.

Case 5: $t_1 \in T^1_M$ and $(\exists w)(w \in T^2_I)$ and $w = \{s_{1,\ldots,} s_k\}$ and $t_2 \in w$. Similar to Case 4, $t \in v^o$.

Therefore, $M(<T_1 \infty T_2>) = v^o$.

Similarly, we can prove that $v^o = M(<T_1 \infty T_2>)$.

Therefore, $v = v'$. □.

Theorem 3.3.1: The number of pair-up operations required by Algorithm 3.3.1 grows exponentially with respect to the size of the underlying I-tables.

Proof: Let $T_D^1 = \{t_1^1,...,t_{N_{D_1}}^1\}, T_I^1 = \{w_1^1,...,w_{N_{I_1}}^1\}$, where $w_i^1 = \{t_{i1}^1,...,t_{ik_i^1}^1\}$, $1 \le i \le N_{I_1}$, and $T_M^1 = \{t_1,...,t_{N_{M_1}}\}$. Let $T_D^2 = \{t_1^2,...,t_{N_{D_2}}^2\}$, $T_I^2 = \{w_1^2,...,w_{N_{I_2}}^2\}$, where $w_i^2 = \{t_{i1}^2,...,t_{ik_i^2}^2\}$, $1 \le i \le N_{I_2}$, and $T_M^2 = \{t_1,...,t_{N_{M_2}}\}$. Then the number of comparisons required by each step of Algorithm 3.3.1 is as follows:

Step 1: 0 comparison,

Step 2: $(\prod_{n=1}^{N_{I_1}} k_n^1 \times N_{D_2} + \prod_{n=1}^{N_{I_2}} k_n^2 \times N_{D_1} + \prod_{n=1}^{N_{I_1}} k_n^1 \times \prod_{n=1}^{N_{I_2}} k_n^2)$ comparisons,

Step 3:

Step 3.1: $(N_{D_1} \times N_{D_2})$ comparisons,

Step 3.2: 0 comparison, and

Step 3.3: $(N_{D_1} \times N_{M_2} + N_{D_2} \times N_{M_1} + N_{M_1} \times N_{M_2}) + (\prod_{n=1}^{N_{I_1}} k_n^1) \times N_M^2 + (\prod_{n=1}^{N_{I_2}} k_n^2) \times N_M^1$

comparisons.

Let K_1 be the largest k_i^1 and K_2 be the largest k_i^2. The total number of comparisons is then of:

$$O(K_1^{N_{I_1}} \times N_{D_2} + K_2^{N_{I_2}} \times N_{D_2} + K_1^{N_{I_1}} \times K_2^{N_{I_2}} +$$
$$N_{D_1} \times N_{D_2} + N_{D_1} \times N_{M_2} + N_{D_2} \times N_{M_1} + N_{M_1} \times N_{M_2} + K_1^{N_{I_1}} \times N_M^2 + K_2^{N_{I_2}} \times N_M^1)$$

which grows exponentially with respect to the size of T_I. □

Theorem 3.3.2: The number of I/O block accesses required by Algorithm 3.3.1 grows

exponentially with respect to the size of the underlying I-tables.

Proof: Let $T_D^1 = \{t_1^1,...,t_{N_{D_1}}^1\}, T_I^1 = \{w_1^1,...,w_{N_{I_1}}^1\}$, where $w_i^1 = \{t_{i1}^1,...,t_{ik_i^1}^1\}$, $1 \le i \le N_{I_1}$, and $T_M^1 = \{t_1,...,t_{N_{M_1}}\}$. Let $T_D^2 = \{t_1^2,...,t_{N_{D_2}}^2\}$, $T_I^2 = \{w_1^2,...,w_{N_{I_2}}^2\}$, where $w_i^2 = \{t_{i1}^2,...,t_{ik_i^2}^2\}$, $1 \le i \le N_{I_2}$, and $T_M^2 = \{t_1,...,t_{N_{M_2}}\}$.

It is sufficient to show that the number of I/O block accesses required to obtain A_{ij} grows exponentially with respect to the size of the underlying I-tables. Note that E has $\prod_{i=1}^{N_{I_1}} k_i^1$ elements and E has $\prod_{i=1}^{N_{I_2}} k_i^2$ elements. Let k_1 be the smallest k_i^1 and k_2 be the smallest k_i^2. Then E contains at least $k_1^{N_{I_1}}$ elements and F contains at least $k_2^{N_{I_2}}$. Suppose that the main memory can hold m blocks while n_1 records of T_1 and n_2 records of T_2 respectively fit on one main memory block. Then the total number of block accesses required to produce A_{ij} is at least:

$$\left\lceil \frac{N_{D_1}}{n_1}\left(1+\frac{k_2^{N_{I_2}}}{(m-1)\times n_2}\right)\right\rceil + \left\lceil \frac{k_2^{N_{I_2}}}{n_2}+\left(1+\frac{k_1^{N_{I_1}}}{(m-1)\times n_1}\right)\right\rceil + \left\lceil \frac{N_{D_2}}{n_2}\left(1+\frac{k_1^{N_{I_1}}}{(m-1)\times n_1}\right)\right\rceil$$

which grows exponentially with respect to the size of I-tables. □

Theorem 3.3.3: The number of pair-up operations required by Algorithm 3.3.2 grows linearly in general with respect to the size of the underlying I-tables and polynomially in the worst case with respect to the size of the underlying I-table.

Proof: Let $T_D^1 = \{t_1^1,...,t_{N_{D_1}}^1\}, T_I^1 = \{w_1^1,...,w_{N_{I_1}}^1\}$, where $w_i^1 = \{t_{i1}^1,...,t_{ik_i^1}^1\}$, $1 \le i \le N_{I_1}$, and $T_M^1 = \{t_1,...,t_{N_{M_1}}\}$. Let $T_D^2 = \{t_1^2,...,t_{N_{D_2}}^2\}$, $T_I^2 = \{w_1^2,...,w_{N_{I_2}}^2\}$, where $w_i^2 = \{t_{i1}^2,...,t_{ik_i^2}^2\}$, $1 \le i \le N_{I_2}$, and $T_M^2 = \{t_1,...,t_{N_{M_2}}\}$.

The worst scenario that could happen is that almost all of the **A** values in the tuples of one of the I-tables occur in the **A** values of the other I-table. The number of pari-up operations of

Algorithm 3.3.2 in this case can be summarized as follows:

Step 1: the number of pari-up operations, in the worst case, necessary for obtaining:

- **ID** is $(k_1^1 + \cdots + k_{N_{I_1}}^1) \times N_{D_2}$,
- **IM** is $(k_1^1 + \cdots + k_{N_{I_1}}^1) \times N_{M_2}$,
- **DI** is $(k_1^2 + ... + k_{N_{I_2}}^2) \times N_{D_1}$,
- **MI** is $(k_1^2 + ... + k_{N_{I_2}}^2) \times N_{M_1}$,
- **II** is $(k_1^1 + ... + k_{N_{I_1}}^1) \times (k_1^2 + \cdots + k_{N_{I_2}}^2)$

Step 2: 0.

Step 3:

Step 3.1: $N_{D_1} \times N_{D_2}$,

Step 3.2: 0, and

Step 3.3: $(N_{D_1} * N_{M_2} + N_{D_2} * N_{M_1} + N_{M_1} * N_{M_2})$

Therefore, the number of pair-up operations grows, in this case, polynomially with respect to the size of the underlying I-tables.

On the other hand, assume that the value occurrences of **A** in both T_1 and T_2 are uniformly distributed over the domain of **A** and the attribute set **A** is a composite key (this scenario can be viewed as the best practical case), then the number of pari-up operations of Algorithm 3.3.2 can be analyzed as follows:

Step 1: the number of pari-up operations necessary for obtaining **ID**, **IM**, **DI**, **MI**, and **II** are as follows:

since **A** is a key, then it can be assumed that the **A** values of the tuples of T_D^2 are distinct. Suppose that all the **A** values in the tuples of T_D^2 appear in the **A** values of the tuples of T_I^1, then the maximum number of pair-up operations needed to construct **ID**

is of $O(k_1^1 + k_2^1 + ... + k_{N_{I_1}}^1)$.

Similarly, the maximum number of pair-up operations needed to form **DI** *is of* $O(k_1^2 + k_2^2 + ... + k_{N_{I_2}}^1)$.

The above analysis can be applied to **IM** and **MI** as well. Hence, the maximum number of pair-up operations required to compute **IM** is of $O(k_1^1 + k_2^1 + ... + k_{N_{I_1}}^1)$ and that of **MI** is of $O(k_1^2 + k_2^2 + ... + k_{N_{I_2}}^1)$.

Although the attribute set **A** is a composite key, its values in the tuples of T_I^1 and T_I^2 cannot be assumed to be distinct due to the disjunctive nature of the indefinite component of I-tables. However, since the value occurrences of **A** in both T_1 and T_2 are uniformly distributed over the domain of **A**, it is then assumed that each **A** value of the tuples of T_I^1 and T_I^2 repeats itself within its own tuple set. The worst case is that every **A** value of the tuples of T_I^1 matches the **A** values of the tuples of T_I^2. Consequently, the maximum number of pair-up operations necessary for obtaining **II** is of $O(K_1 \times K_2 \times N_{I_2})$, where K_1 is the largest k_i^1, $1 \le i \le N_{I_1}$ and K_2 is the largest k_i^2, $1 \le i \le N_{I_2}$. Note that both K_1 and K_2 are usually small constants.

Step 2: 0.

Step 3:

Step 3.1: since **A** is a key, then it can be assumed that the **A** values of the tuples of T_D^1 and T_D^2 are distinct. Suppose that all the **A** values in the tuples of T_D^2 appear in the **A** values of the tuples of T_D^1, then the maximum number of pair-up operations needed to construct **ID** is of $min(N_{D_1}, N_{D_2})$, where $min(x,y)$ is a function which returns the smaller of x and y,

Step 3.2: 0, and

Step 3.3: $min(N_{D_1}, N_{M_2}) + min(N_{D_2}, N_{M_1}) + min(N_{M_1} * N_{M_2})$.

Therefore, in this case, the number of pair-up operations grows linearly with respect to the size of the underlying I-tables. □

Theorem 3.3.4: The number of I/O block accesses required by Algorithm 3.3.2 grows exponentially with respect to the size of the underlying I-tables.

Proof: It is sufficient to note that A_{ij} contains exponential number of elements and Algorithm 3.3.2 still physically constructs A_{ij}, similar to Algorithm 3.3.1. Therefore, the number of I/O block accesses required by Algorithm 3.3.2 still grows exponentially with respect to the the size of the underlying I-tables. □

Theorem 3.3.5: The number of I/O block accesses required by Algorithm 3.3.2 grows linearly with respect to the size of the underlying I-tables, if virtual relations are used for intermediate results.

Proof: The exponential complexity of the block accesses of Algorithm 3.3.2 is due to the fact that all the intermediate relations (e.g., *EE*, *FF*, *EF*'s, and A_{ij}'s) are actually created. Since virtual relations eliminates the need for creating these intermediate relations, the number of I/O block accesses will be reduced to a linear order of complexity with respect to the size of the underlying I-tables. □

Theorem 4.4.1: Let R be a relational scheme and T be a reduced I-table over R, then:

(1) $\mathbf{INS}_D^{\Sigma} < t > (REP(T)) = REP(\mathbf{INS}_D^{\Gamma} < t > (T))$,

(2) $\mathbf{INS}_I^{\Sigma} < \{t_1, t_2, \cdots, t_n\} > (REP(T)) = REP(\mathbf{INS}_{I_u}^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (T))$,

(3) $\mathbf{INS}_M^{\Sigma} < t > (REP(T)) = REP(\mathbf{INS}_M^{\Gamma} < t > (T))$,

(4) $\mathbf{DEL}_D^{\Sigma} < t > (REP(T)) = REP(\mathbf{DEL}_D^{\Gamma} < t > (T))$,

(5) $\mathbf{DEL}_I^{\Sigma}<\{t_1,t_2,\cdots,t_n\}>(REP(T))=REP(\mathbf{DEL}_{I_u}^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))$,

(6) $\mathbf{DEL}_M^{\Sigma}<t>(REP(T))=REP(\mathbf{DEL}_M^{\Gamma}<t>(T))$,

(7) $\mathbf{INS}_I^{\Sigma}<t,\{t_1,t_2,\cdots,t_n\}>(<U,v>)=REP(\mathbf{INS}_{I_t}^{\Gamma}<t,\{t_1,t_2,\cdots,t_n\}>(T))$,

(8) $\mathbf{DEL}_I^{\Sigma}<t_i,\{t_1,\ldots,t_i,\cdots,t_n\}>(<U,v>)=$

$REP(\mathbf{DEL}_{I_t}^{\Gamma}<t_i,\{t_1,\ldots,t_i,\cdots,t_n\}>(T))$.

Proof:

It has been proven in [LiSu90] that $REP(REDUCE(T))=REP(T)$ for any I-table T. This result will be used in the following proofs.

(1) $\mathbf{INS}_D^{\Sigma}<t>(REP(T))$

$=REDUCEREP<\{r\cup\{t\}\,|\,r\in U\},v>$

and $OCCUR(r\cup\{t\},t)=1$ ·

$=REDUCEREP<\{r\cup\{t\}\,|\,r\in\{T_D\cup\{t_1,\cdots,t_n\}\,|\,(\forall i)(1\le i\le n\rightarrow t_i\in w_i)\}\},v>$

and $OCCUR(r\cup\{t\},t)=1$

$=REDUCEREP<\{r\,|\,r\in\{(T_D\cup\{t\})\cup\{t_1,\cdots,t_n\}\,|\,(\forall i)(1\le i\le n\rightarrow t_i\in w_i)\}\},v>$

and $OCCUR(r\cup\{t\},t)=1$

$=REP(REDUCE(\mathbf{INS}_D^{\Gamma}<t>(T)))$

$=REP(\mathbf{INS}_D^{\Gamma}<t>(T))$.

Therefore, $\mathbf{INS}_D^{\Sigma}<t>(REP(T))=REP(\mathbf{INS}_D^{\Gamma}<t>(T))$

(2) $\mathbf{INS}_I^{\Sigma}<\{t_1,t_2,\cdots,t_n\}>(REP(T))$

$=REDUCEREP<\{r\cup\{t\}\,|\,r\in U\wedge t\in\{t_1,t_2,\cdots,t_n\}\},v>$ and

$OCCUR(r\cup\{t\},t)=OCCUR(r,t)+1$

$=REDUCEREP<\{r\cup\{t\}\,|\,r\in\{T_D\cup\{t'_1,\cdots,t'_m\}\,|\,(\forall i)(1\le i\le m\rightarrow t'_i\in w_i)\}\wedge$

$t\in\{t_1,t_2,\cdots,t_n\}\},v>$ and

$OCCUR(r\cup\{t\},t)=OCCUR(r,t)+1$

$$=REDUCEREP<\{r|r\in\{T_D\cup\{t'_1,\cdots,t'_m\}\cup\{t\}|(\forall i)(1\le i\le m\rightarrow t'_i\in w_i)\}\wedge$$
$$t\in\{t_1,t_2,\cdots,t_n\}\},v>\text{ and}$$
$$OCCUR(r\cup\{t\},t)=OCCUR(r,t)+1$$
$$=REDUCEREP<\{r|(r\in\{T_D\cup\{t'_1,\cdots,t'_m,t\}|(\forall i)(0\le i\le m+1\rightarrow t'_i\in w_i)\})\wedge$$
$$w_{m+1}=\{t_1,t_2,\cdots,t_n\}\}\},v>\text{ and}$$
$$OCCUR(r\cup\{t\},t)=OCCUR(r,t)+1$$
$$=REP(REDUCE(\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T)))$$
$$=REP(\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))$$

Therefore, $\mathbf{INS}_I^{\Sigma}<\{t_1,t_2,\cdots,t_n\}>(REP(T))=REP(\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))$,

(3) Apparently, by definitions of $\mathbf{INS}_M^{\Sigma}$, $\mathbf{INS}_M^{\Gamma}$, and REP, $U_1=U_2$ and $v_1=v_2=v\cup\{t\}$.

Therefore, $\mathbf{DEL}_M^{\Sigma}<t>(REP(T))=REP(\mathbf{INS}_M^{\Gamma_M}<t>(T))$

(4) $\mathbf{DEL}_D^{\Sigma}<t>(REP(T))$

$$=REDUCEREP<\{r-t\cap(\bigcap_{r_1\in U}r_1)|r\in U\},v>\text{ and}$$
$$OCCUR(r-\{t\},t)=0$$
$$=REDUCEREP<\{r-t\cap(\bigcap_{r_1\in U}r_1)|$$
$$r\in\{T_D\cup\{t_1,\cdots,t_n\}|(\forall i)(1\le i\le n\rightarrow t_i\in w_i)\},v>,\text{ and}$$
$$OCCUR(r-\{t\},t)=0$$
$$=REDUCEREP<\{r|(r\in\{(T_D-t\cap(\bigcap_{r_1\in U}r_1))\cup\{t_1,\cdots,t_n\}|$$
$$(\forall i)(1\le i\le n\rightarrow t_i\in w_i)\},v>,\text{ and}$$
$$OCCUR(r-\{t\},t)=0$$
$$=REP(REDUCE(\mathbf{DEL}_D^{\Gamma}<t>(T))).$$
$$=REP(\mathbf{DEL}_D^{\Gamma}<t>(T)).$$

Therefore, $\mathbf{DEL}_D^{\Sigma}<t>(REP(T))=REP(\mathbf{DEL}_D^{\Gamma}<t>(T))$

(5) Let $REP(T)=<U,v>$, $\mathbf{DEL}_I^{\Sigma}<\{t_1,t_2,\cdots,t_n\}>(REP(T))=<U_1,v_1>$, and $REP(\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))=<U_2,v_2>$. There are two possible cases:

Case 1: none of the tuples of $\{t_1,t_2,\cdots,t_n\}$ appears in any other tuple set of T_I. In this case, all the t_i's, $1\leq i\leq n$ will be removed from every definite relation of U and their corresponding *OCCUR*'s will be set to 0 (i.e., does not exist). That is, the possibilities that each t_i's, $1\leq i\leq n$, is a part of the real world truth is no longer valid. Let r be an arbitrary relation of U_1, this implies that there is a r_1 of U and a t_i, $1\leq i\leq n$, such that $r=r_1-t_i$. Clearly, r is a member of U_2. On the other hand, $v_1=v_2=v$. Hence, $<U_1,v_1>\subseteq<U_2,v_2>$. Similarly, it can be proven that $<U_2,v_2>\subseteq<U_1,v_1>$.

Case 2: some (but not all) of the t_i's, $1\leq i\leq n$ and $n\geq 3$, of $\{t_1,\ldots,t_n\}$ also appears in some other tuple set of T_I. Suppose that t of $\{t_1,\ldots,t_n\}$ also appears in some other tuple set of T_I, then $OCCUR(r,t)$, $r\in U$, must be greater than 1 if $t\in r$. r can then be thought of containing $OCCUR(r,t)$ number of t's. Therefore, the arguments of case 1 applies to this case as well.

Therefore,

$$\mathbf{DEL}_I^{\Sigma}<\{t_1,t_2,\cdots,t_n\}>(REP(T))=REP(\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(REP(T)))$$

(6) Apparently, by definitions of $\mathbf{INS}_M^{\Sigma}$, $\mathbf{INS}_M^{\Gamma}$, and *REP*, $U_1=U_2$ and $v_1=v_2=v-\{t\}$.

Therefore, $\mathbf{DEL}_M^{\Sigma}<t>(REP(T))=REP(\mathbf{DEL}_M^{\Gamma}<t>(T))$

(7) from (2) and (5).

(8) from (2) and (5). □

Theorem 4.4.2: Let R be a relational scheme and T be a reduced I-table over R, then:

(1) $\mathbf{DEL}_D^{\Gamma}<t>(\mathbf{INS}_D^{\Gamma}<t>(T))\neq T$

(2) $\mathbf{INS}_D^{\Gamma}<t>(\mathbf{DEL}_D^{\Gamma}<t>(T))=T$

(3) $\mathbf{DEL}_D^{\Gamma}<t>(\mathbf{INS}_D^{\Gamma}<t>(T))\neq\mathbf{INS}_D^{\Gamma}<t>(\mathbf{DEL}_D^{\Gamma}<t>(T))$

(4) $\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))\neq T$

(5) $\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))=T$

(6) $\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))$

$\neq\mathbf{INS}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(\mathbf{DEL}_I^{\Gamma}<\{t_1,t_2,\cdots,t_n\}>(T))$

(7) $\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_n\}>(\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_n\}>(T))\neq T$

(8) $\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t_{m+1},\cdots,t_n\}>$

$(\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t,t_{m+1},\cdots,t_n\}>(T))=T$

(9) $\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t_{m+1},\cdots,t_n\}>$

$(\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t,t_{m+1},\cdots,t_n\}>(T))\neq$

$\mathbf{DEL}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t_{m+1},\cdots,t_n\}>$

$(\mathbf{INS}_I^{\Gamma}<t,\{t_1,t_2,\cdots,t_m,t,t_{m+1},\cdots,t_n\}>(T))$

(10) $\mathbf{DEL}_M^{\Gamma}<t>(\mathbf{INS}_M^{\Gamma}<t>(T))=\mathbf{INS}_M^{\Gamma}<t>(\mathbf{DEL}_M^{\Gamma}<t>(T))=T$

Proof: Since (1) $\wedge$ (2) $\rightarrow$ (3), (4) $\wedge$ (5) $\rightarrow$ (6), and (7) $\wedge$ (8) $\rightarrow$ (9), in addition, we will omit the proof for (10) since it is trivial, we then only need to prove (1), (2), (4), (5), (7), and (8).

We will use the set notation to represent an I-table. We denote an I-table T over R with $\{T_D,T_I,T_M\}$, where T_D is itself a set containing tuples of R, T_I is a set of tuple set with each tuple set containing tuples of R, and T_M is also a set containing tuples of R.

(1) Disprove by a counter example: Let $T=\{\{a\},\ \{\{b,c\}\},\varnothing\}$,

then $\mathbf{INS}_D^{\Gamma}<b>(T)=\{\{a,b\},\varnothing,\{c\}\}$ and

$\mathbf{DEL}_D^{\Gamma}<b>(\mathbf{INS}_D^{\Gamma}<b>(T))=\mathbf{DEL}_D^{\Gamma}<b>(\{\{a,b\},\varnothing,\{c\}\})=\{\{a\},\varnothing,\{c\}\}\neq T$

(2) $\mathbf{INS}_D^{\Gamma}<t>(\mathbf{DEL}_D^{\Gamma}<t>(T))$

$=\mathbf{INS}_D^{\Gamma}<t>(\mathbf{DEL}_D^{\Gamma}<t>(\{T_D,T_I,T_M\}))$

$=\mathbf{INS}_D^{\Gamma}<t>(\{T_D-\{t\},T_I,T_M\})$

$=REDUCE(\{(T_D-\{t\})\cup\{t\},T_I,T_M\}))$

$= \{T_D, T_I, T_M\}$ (since T is reduced)

$= T$

(4) Disprove by a counter example: Let $T = \{\varnothing, \{\{a,b,c\}\}, \varnothing\}$,

then $\mathbf{INS}_I^{\Gamma} < \{a,b\} > (T) = \{\varnothing, \{\{a,b\}\}, \{c\}\}$ and

$\mathbf{DEL}_D^{\Gamma} < \{a,b\} > (\mathbf{INS}_D^{\Gamma} < \{a,b\} > (T)) =$

$\mathbf{DEL}_D^{\Gamma} < \{a,b\} > (\{\varnothing,, \varnothing, \{c\}\}) = \{\varnothing, \varnothing, \{c\}\} \neq T$

(5) $\mathbf{INS}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (\mathbf{DEL}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (T))$

$= \mathbf{INS}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (\mathbf{DEL}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (\{T_D, T_I, T_M\}))$

$= \mathbf{INS}_I^{\Gamma} < \{t_1, t_2, \cdots, t_n\} > (\{T_D, T_I - \{t_1, t_2, \cdots, t_n\}, T_M\})$

$= REDUCE(\{T_D, (T_I - \{t_1, t_2, \cdots, t_n\}) \cup \{t_1, t_2, \cdots, t_n\}, T_M\})$

$= \{T_D, T_I, T_M\}$ (since T is reduced)

$= T$

(7) Disprove by a counter example: Let $T = \{\varnothing, \{\{a,b\}, \{a,c\}\}, \varnothing\}$,

then $\mathbf{INS}_I^{\Gamma} < b, \{a,c\} > (T) = \{\varnothing, \{\{a,b\}\}, \{c\}\}$ and

$\mathbf{DEL}_D^{\Gamma} < b, \{a,c\} > (\mathbf{INS}_D^{\Gamma} < b, \{a,c\} > (T)) = \mathbf{DEL}_D^{\Gamma} < b, \{a,c\} > (\{\varnothing, \{\{a,b\}\}, \{c\}\})$

$= \{\varnothing, \{\{a,b\}\}, \{c\}\} \neq T$

(8) $\mathbf{INS}_I^{\Gamma} < t, \{t_1, t_2, \cdots, t_m, t_{m+1}, \cdots, t_n\} >$

$(\mathbf{DEL}_I^{\Gamma} < t, \{t_1, t_2, \cdots, t_m, t, t_{m+1}, \cdots, t_n\} > (T)) =$

$\mathbf{INS}_I^{\Gamma} < t, \{t_1, t_2, \cdots, t_m, t_{m+1}, \cdots, t_n\} >$

$(\mathbf{DEL}_I^{\Gamma} < t, \{t_1, t_2, \cdots, t_m, t, t_{m+1}, \cdots, t_n\} > (\{T_D, T_I, T_M\}))$

$= \mathbf{INS}_I^{\Gamma} < t, \{t_1, t_2, \cdots, t_m, t_{m+1}, \cdots, t_n\} >$

$(\{T_D, (T_I - \{t_1, t_2, \cdots, t_m, t, t_{m+1}, \cdots, t_n\}) \cup$

$\{t_1, t_2, \cdots, t_m, t_{m+1}, \cdots, t_n\}, T_M\})$

$= REDUCE(\{T_D, ((T_I - \{t_1, t_2, \cdots, t_m, t, t_{m+1}, \cdots, t_n\}) \cup$

$\{t_1, t_2, \cdots, t_m, t_{m+1}, \cdots, t_n\}) - i$

$$\{t_1,t_2,\cdots,t_m,t_{m+1},\cdots,t_n\})\cup\{t_1,t_2,\cdots,t_m,t,t_{m+1},\cdots,t_n\}),T_M\})$$

$= \{T_D,T_I,T_M\}$ (since T is reduced)

$= T$ □

Theorem 6.3.1 $a \wedge (b \mid c) =_E (a \wedge b) \mid (a \wedge c)$.

Proof: The equivalence can be proven by the following truth table:

a	b	c	$a \wedge (b \mid c)$	$(a \wedge b) \mid (a \wedge c)$
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	u	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	u	u

□

Theorem 6.3.2 The following equivalence holds:

$$a_1 \mid a_2 \mid \cdots \mid a_n =_E (a_1 \mid a_2 \mid \cdots \mid a_{n-1}) \mid a_n$$

Proof: We prove by induction on n.

(1) Basis: let $n=3$, then $a_1 \mid a_2 \mid a_3 =_E (a_1 \mid a_2) \mid a_3$ as shown by the following truth table:

a	b	c	$a \mid b \mid c$	$a \mid b$	$(a \mid b) \mid c$
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	1	1	1
0	1	1	u	1	u
1	0	0	1	1	1
1	0	1	u	1	u
1	1	0	u	u	u
1	1	1	u	u	u

(2) Induction Step: suppose that $a_1 \mid \cdots \mid a_n =_E (a_1 \mid \cdots \mid a_{n-1}) \mid a_n$ for an arbitrary n such that $n > 3$, then

$a_1 \mid \cdots \mid a_n \mid a_{n+1}$

$=_E (a_1 \mid \cdots \mid a_{n-1}) \mid a_n \mid a_{n+1}$ *(by Induction Hypothesis)*

$=_E ((a_1 \mid \cdots \mid a_{n-1}) \mid a_n) \mid a_{n+1}$ *(by Basis)*

$=_E (a_1 \mid \cdots \mid a_{n-1} \mid a_n) \mid a_{n+1}$ *(by Basis).* □

Theorem 6.3.3 $a \wedge (a \mid B_1 \mid \ldots\ldots \mid B_n) =_E a$, where a is an atomic ground formula representing a singleton tuple and B_i, $1 \leq i \leq n$, is a conjunction of ground formulas which are free of a's.

Proof: We prove by induction on n, that is, the number of B_i's.

(1) Basis: let $n = 1$, then $a \wedge (a \mid B) =_E a$ as shown by the the following truth table:

a	B	$a \wedge (a \mid B)$
0	0	0
0	1	0
1	0	1
1	1	u

(2) Induction Step: suppose that $a \wedge (a \mid b_B \mid \ldots\ldots \mid B_n) =_E a$ for an arbitrary n such that $n > 1$, then

$a \wedge (a \mid B_1 \mid \ldots\ldots \mid B_n \mid B_{n+1})$

$=_E a \wedge ((a \mid B_1 \mid \ldots\ldots \mid B_n) \mid B_{n+1})$ (*by Theorem*6.3.2)

$=_E (a \wedge (a \mid B_1 \mid \ldots\ldots \mid B_n)) \mid (a \wedge B_{n+1})$ (*by Theorem* 6.3.1)

$=_E a \mid (a \wedge B_{n=1})$ (*by induction hypothesis*)

$=_E a \wedge (a \mid B_{n+1})$ (*by Theorem* 6.3.1)

$=_E a$ (*by the Basis Step*) □

Theorem 6.3.4 $A \wedge (A \mid B_1 \mid \ldots\ldots \mid B_n) =_E A$, where A is a conjunction of atomic ground formulas and each conjunct can be a generalized exclusive disjunction (e.g., $A = a \wedge (b \mid c)$). Each B_i, $1 \leq i \leq n$, is a conjunction of ground formulas which does not contain the elements of A.

Proof: From Theorem 6.3.3.

Theorem 6.3.5 The following equivalence holds:

$a \wedge ((a \wedge b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m) =_E$

$a \wedge ((b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m)$

where a and b_i's, $1 \leq i \leq n$ are atomic ground formulas and Each C_i, $1 \leq i \leq n$, is a conjunction of ground formulas which are free of a and b_i's, $1 \leq i \leq n$.

Proof: We prove by induction on m, that is, the number of C_i's.

(1) Basis: let $m = 1$, then

$a \wedge ((a \wedge b_1 \wedge \cdots \wedge b_n) \mid C_1)$

$=_E (a \wedge a \wedge b_1 \wedge \cdots \wedge b_n) \mid (a \wedge C_1)$ (*by Theorem*6.3.1)

$=_E (a \wedge b_1 \wedge \cdots \wedge b_n) \mid (a \wedge C_1)$

$=_E a \wedge ((b_1 \wedge \cdots \wedge b_n) \mid C_1)$ (*by Theorem* 6.3.1)

(2) Induction Step: suppose that

$a \wedge ((a \wedge b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m) =_E$

$a \wedge ((b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m)$ for an arbitrary m such that $m > 1$, then

$a \wedge ((a \wedge b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m \mid C_{m+1})$

$=_E a \wedge (((a \wedge b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m) \mid C_{m+1})$ (*by Theorem* 6.3.2)

$=_E a \wedge ((a \wedge (b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m)) \mid (a \wedge C_{m+1})$

(*by Theorem* 6.3.1)

$=_E a \wedge ((b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m) \mid (a \wedge C_{m+1})$

(*by Induction Hypothesis*)

$=_E a \wedge (((b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m) \mid C_{m+1})$ (*by Theorem* 6.3.1)

$=_E a \wedge ((b_1 \wedge \cdots \wedge b_n) \mid C_1 \mid \ldots\ldots \mid C_m \mid C_{m+1})$ (*by Theorem* 6.3.2) □

Theorem 6.3.6 The following equivalence holds:

$(a \wedge b_1) \mid (a \wedge b_2) \mid \ldots\ldots \mid (a \wedge b_n) =_E a \wedge (b_1 \mid \ldots\ldots \mid b_n)$

in 0 where Each b_i, $1 \leq i \leq n$, is a conjunction of ground formulas.

Proof: We prove by induction on n, that is, the number of b_i's.

(1) Basis: let $n = 1$, then by Theorem 6.3.1, $(a \wedge b_1) \mid (a \wedge b_2) =_E a \wedge (b_1 \mid b_2)$

(2) Induction Step: suppose that

$(a \wedge b_1) \mid (a \wedge b_2) \mid \ldots\ldots \mid (a \wedge b_n) = a \wedge (b_1 \mid \ldots\ldots \mid b_n)$ for an arbitrary n such that $n > 2$, then

$(a \wedge b_1) \mid (a \wedge b_2) \mid \ldots\ldots \mid (a \wedge b_n) \mid (a \wedge b_{n+1})$

$=_E ((a \wedge b_1) \mid (a \wedge b_2) \mid \ldots\ldots \mid (a \wedge b_n)) \mid (a \wedge b_{n+1})$ (*by Theorem* 6.3.2)

$=_E (a \wedge (b_1 \mid \ldots\ldots \mid b_n)) \mid (a \wedge b_{n+1})$ (*by Induction Hypothesis*)

$=_E a \wedge ((b_1 \mid \ldots\ldots \mid b_n) \mid b_{n+1})$ (*by Theorem* 6.3.1)

$=_E a \wedge (b_1 \mid \ldots\ldots \mid b_n \mid b_{n+1})$ (*by Theorem* 6.3.2) □

Theorem 6.3.7 The following equivalence holds:

$$(a_1 \wedge \ldots\ldots \wedge a_n) \mid (a_1 \wedge \ldots\ldots \wedge a_n \wedge B) =_E (a_1 \wedge \ldots\ldots \wedge a_n)$$

where B is a conjunction of ground atomic formulas which is free of a_i's, $1 \leq i \leq n$.

Proof: We prove by induction on n, that is, the number of a_i's.

(1) Basis: let $n = 1$, then $a \mid (a \wedge B) =_E a$ according to the following truth table:

a	B	$a \wedge B$	$a \mid (a \wedge B)$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	u

(2) Induction Step: suppose that

$(a_1 \wedge \ldots\ldots \wedge a_n) \mid (a_1 \wedge \ldots\ldots \wedge a_n \wedge B) =_E (a_1 \wedge \ldots\ldots \wedge a_n)$ for an arbitrary n such that $n > 1$,

then

$$(a_1 \wedge \ldots\ldots \wedge a_n \wedge a_{n+1}) \mid (a_1 \wedge \ldots\ldots \wedge a_n \wedge a_{n+1} \wedge B)$$

$$=_E ((a_1 \wedge \ldots\ldots \wedge a_n) \wedge (a_{n+1})) \mid ((a_1 \wedge \ldots\ldots \wedge a_n \wedge B) \wedge (a_{n+1}))$$

$$=_E ((a_1 \wedge \ldots\ldots \wedge a_n) \mid (a_1 \wedge \ldots\ldots \wedge a_n \wedge B)) \wedge a_{n+1} \qquad (\textit{by Theorem } 6.3.1)$$

$$=_E (a_1 \wedge \ldots\ldots \wedge a_n) \wedge (a_{n+1}) \qquad (\textit{by Induction Hypothesis})$$

$$=_E a_1 \wedge \ldots\ldots \wedge a_n \wedge a_{n+1} \qquad \square$$

Theorem 6.3.8: For any E-table $T \in \Gamma_R$, $REDUCE(REDUCE(T)) = REDUCE(T)$.

Proof: Follows from the definition of *REDUCE*. $\square$

Theorem 6.3.9: For any E-tables $T_1 \in \Gamma_R$ and $T_2 \in \Gamma_R$, if $T_1 =_E T_2$ then $REP(T_1) = REP(T_2)$.

Proof: Follows from the definition of *REP*. $\square$

Theorem 6.3.10: For any E-table $T \in \Gamma_R$, $REPREP(REP(REDUCE(T))) = REPREP(REP(T))$.

Proof: We prove for each of the nine types of redundancies:

$REPREP(REP(REDUCE(T))) = REPREP(REP(T))$ for any E-table T.

Case 1: T contains the first type of redundancies:

(1) if the set of tuple sets to be reduced is not a part of any entanglement, then by Theorem 6.3.3 and Theorem 6.3.9,

$REPREP(REP(REDUCEREP(T))) = REPREP(REP(T))$

(2) $T = <T_D, T_E>$ contains entanglements: then there exists a tuple set w of some set of tuple sets W of T_E and w is a subset of T_D. Further, the tuple sets of $W - w$ entangle the indefinite component. Let

$T_E = \{ W_1, \ldots, W_k, W_{k+1}, \ldots, W_{k+K}, W_{k+K+1}, \ldots, W_n \}$, where

(i) $W_1 = \{ w_{11}, w_{12}, \ldots, w_{1m_1} \}$ such that w_{11} is a subset of T_D, and

(ii) $W_{k+i} = \{ w_{(k+i)1}, \ldots, w_{(k+i)k_i}, w_{(k+i)(k_i+1)}, \ldots, w_{(k+i)m_{(k+i)}} \}$, for $1 \le i \le K$, such that $w_{(k+i)j}$'s, for $1 \le j \le k_i$, contain the tuples of $W_1 - w_{11}$ and $w_{(k+i)l}$'s, for $k_i + 1 \le l \le m_{(k+i)}$, is free of any tuples of $W_1 - w_{11}$, and

(iii) W_i, $k + K + 1 \le i \le n$, is not a part of any entanglements and redundancies.

Then $REDUCE(T) = <T_D', T_E'>$, where

$T_D' = T_D$,

$T_E' = \{ W_2, \ldots, W_k, W_{k+1}', \ldots, W_{k+K}', W_{k+K+1}, \ldots, W_n \}$ (if any of the W_{k+i}', $1 \le i \le K$, contains only one tuple set, then it should be included in T_D), and

$W_{k+i}' = \{ w_{(k+i)(k_i+1)}, \ldots, w_{(k+i)m_{(k+i)}} \}$, for $1 \le i \le j$.

Now let us first prove that $REP(T) \subseteq REP(REDUCE(T))$. Let r be an arbitrary relation of $REP(T)$. According to the definition of REP, the following observations can be made:

1. since w_{11} of W_1 is a subset of T_D, only w_{11} can be included in any relation of $REP(T)$,

2. Since $w_{(k+i)j}$'s of W_{k+i}'s, for $j=1 \cdots k_i$ and $i=1,...,J$, contain the tuples of $W_1 - w_{11}$, none of them can be chosen to be included in any relation of $REP(T)$. Consequently, only the tuple sets of $W'_{k+i} = \{w_{(k+i)(k_i+1)}, \ldots, w_{(k+i)m_{(k+i)}}\}$, for $1 \le i \le J$, can be candidates for the relations of $REP(T)$.

Therefore, according to the definition for $REDUCE(T)$, r is also in $REP(REDUCE(T))$.

By using the same reasoning, we can prove that $REP(REDUCE(T)) \subseteq REP(T)$.

Hence, $REPREP(REP(REDUCEREP(T))) = REPREP(REP(T))$.

Case 2: T contains the second type of redundancies:

(1) if the set of tuple sets to be reduced is not a part of any entanglement, then by Theorem 6.3.4 and Theorem 6.3.9,

$REPREP(REP(REDUCEREP(T))) = REPREP(REP(T))$.

(2) $T = <T_D, T_E>$ contain entanglements: the proof is similar to that of Case 1 and is therefore omitted.

Case 3: T contains the third type of redundancies, then by Theorem 6.3.5 and Theorem 6.3.9,

$REPREP(REP(REDUCEREP(T))) = REPREP(REP(T))$

Case 4: T contains the fourth type of redundancies, then by Theorem 6.3.6 and Theorem 6.3.9,

$REPREP(REP(REDUCEREP(T))) = REPREP(REP(T))$

Case 5: T contains the fifth type of redundancies:

(1) if the set of tuple sets to be reduced is not a part of any entanglement, then by Theorem 6.3.7 and Theorem 6.3.9,

$REPREP(REP(REDUCEREP(T))) = REPREP(REP(T))$.

(2) $T = <T_D, T_E>$ contain entanglements: the proof is similar to that of Case 1 and is therefore omitted.

Case 6: T contains the sixth type of redundancies. Let $T = <T_D, T_E>$ with $T_E = \{W_1, \ldots\ldots, W_n\}$. In this case, there exist a W_i, $1 \le i \le n$, such that $W_i = \{w_1, \ldots\ldots, w_n, \varepsilon_1, \ldots\ldots, \varepsilon_m\}$, $m \ge 2$, and

REDUCE(T) is T with $W_i=\{w_1,.....,w_n,\varepsilon\}$. Then for any relation r of *REP*(T), if the tuple set chosen from W_i is w_j, $1\leq j\leq n$, then clearly r is also in *REP*(*REDUCE*(T)). If the tuple set chosen from W_i of T is ε_j, $1\leq j\leq m$, then there is relation r' in *REP*(*REDUCE*(T)) such that $r'=(r-\varepsilon_j\cup\varepsilon_k$. On the other hand, for any relation r of *REP*(*REDUCE*(T)), if the tuple set chosen from W_i of *REDUCE*(T) is w_j, $1\leq j\leq n$, then clearly r is also in *REP*(T). If the tuple set chosen from W_i is ε, then there exists a relation r' of *REP*(T) such that $r'=(r-\varepsilon)\cup\varepsilon_l$, $1\leq l\leq m$. Therefore, by the definition of *REPREP*, *REPREP*(*REP*(*REDUCEREP*(T)))=*REPREP*(*REP*(T)).

Case 7: T contains the seventh type redundancies. The proof is similar to that of Case 6 and is therefore omitted.

Case 8: T contains the eighth type of redundancies. Let $T=<T_D,T_E>$ with $T_E=\{W_1,.....,W_n\}$. In this case, there exists a W_i, $1\leq i\leq n$, such that $W_i=\{w_{i1},.....,w_{ij},.....,w_{ik_i}\}$ with $w_{ij}=\{w_1,.....,w_m,E\}$, where E is a set of special dummy values such that none of its members appear in any other sets of tuple sets of the indefinite component. Then *REDUCE*(T) is T without E in w_{ij} of the set of tuple sets W_i. Let us now prove that *REPREP*(*REP*(*REDUCE*(T)))=*REPREP*(*REP*(T)). Let r be an arbitrary relation of *REP*(T), if the tuple set chosen from W_i is w_l, $1\leq l\leq k_l$ and $l\neq j$, then clearly r is also in *REP*(*REDUCE*(T)). If the tuple set chosen from W_i of T is w_{ij}, $1\leq j\leq m$, then ($r-E$ is also in *REP*(*REDUCE*(T)). On the other hand, for any relation r of *REP*(*REDUCE*(T)), if the tuple set chose from W_i is not w_{ij}, then clearly r is also in *REP*(T). If the tuple set chose from W_i is w_{ij}, then ($r\cup E$) is in *REP*(T). Therefore, By the definition of *REPREP*, *REPREP*(*REP*(*REDUCEREP*(T)))=*REPREP*(*REP*(T)).

Case 9: follows from the definition. □

Theorem 6.4.1.1 *REPREP*(*REP*($\sigma_F(T)$))=*REPREP*(σ_F(*REP*(T))) for any consistent and

reduced E-table T and formula F.

Proof: Let T be an arbitrary consistent and reduced E-table. According to the definition of σ on E-tables, $\sigma_F(T)$ is T with those tuples in T not satisfying F replaced by their corresponding dummy values. Note that the only redundancies that could arise after selections are of types six through eight identified in Section 6.3. Therefore, $REP(\sigma_F(T))$ is $REP(T)$ with those tuples of T not satisfying F replaced some special dummy values. On the other hand, $\sigma_F(REP(T))$ is $REP(T)$ with those tuples not satisfying F removed. Hence, according to the definition of $REPREP$, $REPREP(REP(\sigma_F(T)))=REPREP(\sigma_F(REP(T)))$. □

Theorem 6.4.2.1 $REPREP(REP(\pi_A(T)))=REPREP(\pi_A(REP(T)))$ for any consistent and reduced domain compatible E-tables T.

Proof: Let T be an arbitrary reduced and consistent E-table. Following the definition of projection, let $\pi_A(T)=REDUCE(T')$. First we note that redundancies could arise in T'. Since the projection of a dummy value is still a dummy value, only the first through the fifth and the ninth types of redundancies identified in Section 6.3 could occur. Further, we note that in the result of the projection of T, the following four situations could occur: (1) a tuple s of the definite component is also a set of a set of tuple sets W of the indefinite component and at least one of the tuple sets in $W-s$ entangles the indefinite component, (2) a tuple set w of a set of tuple sets is also a member of another set of tuple set W of the indefinite component and at least one of the tuple sets of $W-w$ entangles the indefinite component, (3) a tuple set of a set of tuple sets entangles the indefinite component of T', and (4) a set of tuple sets of T_E become a singleton set after projection. With the above observations in mind, we now prove that:

$REPREP(REP(\pi_A(T)))=REPREP(\pi_A REP(T))$.

Let $T=<T_D,T_E>$, where $T_E=\{W_1,.....W_n\}$ and $W_i=\{w_{i1},.....,w_{im_i}\}$, $1\le i\le n$. Also let $\pi_A(T)=REDUCE(T')$, where $T'_E=\{W'_1,.....W'_n\}$, and $W'_i=\{w'_{i1},.....,w'_{im_i}\}$, $1\le i\le n$.

Case 1: T' contains the first type of redundancies. There are two possibilities:

(1) There exists W_j', $1 \le j \le n$, such that $w_{jk}' \subseteq \pi_A(T_D')$, $1 \le k \le m_j$. Further, none of the tuple sets of $W_j' - w_{jk}'$ entangles the indefinite component of T'. Therefore, W_j' is to be reduced from T' and $\pi_A(T)$ then is T' with W_j' removed. Clearly, in this case, $REP(\pi_A(T)) \subseteq \pi_A(REP(T))$.

To prove that $\pi_A(REP(T)) \subseteq REP(\pi_A(T))$, let $REP(T) = U_1 \cup U_2$ such that $U_1 \cap U_2 = \varnothing$ and each accumulation in U_1 is constructed by selecting w_{jk} from W_j and each accumulation in U_2 is constructed by selecting a tuple set other than w_{jk} from W_j. Since $w_{jk}' = \pi_A(w_{jk}$ is a subset of $\pi_A(T_D)$, each relation in $\pi_A(U_2)$ is a superset of a relation in $\pi_A(U_1)$. Therefore, by the definition of *REDUCEACCACC*, $\pi_A(REP(T)) = \pi_A(U_1)$. Therefore, $\pi_A(REP(T)) \subseteq REP(\pi_A(T))$.

(2) Some tuple set of a set of tuple sets of T_E' is a subset of the definite component and it also entangles T_E'. Then, let

$T_E' = \{ W_1', W_2', \ldots, W_k', W_{k+1}', \ldots, W_n' \}$, where

(i) $W_1' = \{ w_{11}', w_{12}', \ldots, w_{1m_1}' \}$ such that w_{11}' is a subset of T_D',

(ii) for every $W_i' = \{ w_{i1}', \ldots, w_{ik_i}', w_{ik_{i+1}}', \ldots, w_{im_i}' \}$, for $i = 2, \ldots, k$, the w_{ij}''s, for $1 \le j \le k_i$, contain the tuples of $W_1' - w_{11}'$, and the w_{il}''s, for $k_i + 1 \le l \le m_i$, is free of any tuples of $W_1 - w_{11}$, and

(iii) W_g''s, $k+1 \le g \le n$, are not a part of any entanglement of T_E', nor are they redundant (neither with the definite component nor with the indefinite component or among themselves).

Then $\pi_A(T) = REDUCE(<T_D', T_E'>) = <T_D'', T_E''>$, where

$T_D'' = T_D$,

$T_E'' = \{ W_2'', \ldots, W_k'', W_{k+1}', \ldots, W_n' \}$ (if any of the W_i'''s, $1 \le i \le k$, is a singleton set, then it should be included in T_D''), and

$W_i'' = \{ w'_{ik_{i+1}}, \ldots, w'_{im_i} \}$, for $i = 2,\ldots,k$.

Now let us prove that $REPREP(REP(\pi_A(T))) = REPREP(\pi_A(REP(T)))$. Let $\pi_A(REP(T)) = REDUCEACCACC(U)$ (assuming that U is consistent) and let r be an arbitrary relation of $REP(T)$. According to the definition of *REP*, r can be one of the following:

1. $r_1 = T_D \cup w_{1l} \cup w_{2i_2} \cup \cdots \cup w_{ni_n}$, where $2 \le l \le m_1$ and $1 \le i_j \le m_j$ for $2 \le j \le n$.
   Since $\pi_A(w_{11})$ is a subset of $\pi_A(T_D)$, $\pi_A(r)$ should be reduced by *REDUCEACCACC* because the corresponding relation of $REP(T)$ which has the exact same members as r except containing w_{11} as it choice instead of w_{il}, $1 \le l \le m_1$ is a subset of $\pi_A(r)$. Therefore, in this case, r cannot be a member of $\pi_A(REP(T))$.

2. r contains a tuple set w_{ij} from the set of tuple sets W_i, where $2 \le i \le k$ and $1 \le j \le k_i$. Then there exists $w_{1l} \in W_1$, $2 \le l \le m_1$, such that $\pi_A(w_{1l}) = \pi_A(w_{ij})$. In addition, $\pi_A(w_{11})$ is a sub set of $\pi_A(T_D)$. Then, let r be constructed of w_{ij} from W_i, where $2 \le i \le k$ and $1 \le j \le k_i$. Also, let w_{11} be a member of r. On the other hand, let r_1 be exactly the same as r except it has w_{1l}, $2 \le l \le m_1$ instead of w_{11} as one of its member. It is clear that r_1 uis also a member of $REP(T)$. Therefore, $SET(r) = SET(r_1)$. However, $\{\pi_A(r)\}_A \ne \{\pi_A(r_1)\}_A$. Therefore, both r and r_1 are to be removed by *REDUCEACCACC*. thus, r in this case cannot be a member of $\pi_A(REP(T))$.

3. $r = T_D \cup w_{11} \cup w_{2i_2} \cup \cdots \cup w_{ki_k} \cup w_{(k+1)i_{(k+1)}} \cup \cdots \cup w_{ni_n}$, where $k_j + 1 \le i_j \le m_j$ for $2 \le j \le k$ and $1 \le i_j \le m_j$ for $k+1 \le j \le n$.
   From (i), (ii), and (iii), r in this case is a member of $\pi_A(REP(T))$.

Therefore, only the relations of (3) is in $\pi_A(REP(T))$. This implies that, according to

the definition of *REP* and *REPREP*, $REPREP(REP(\pi_A(T)))=REPREP(\pi_A(REP(T)))$.

Case 2: $\pi_A(T)$ contains the second type of redundancies. The proof for this case is similar to Case 1 and is therefore omitted.

Case 3: $\pi_A(T)$ contains the third type of redundancies. That is, there exists t of T_D' and there exists W_j', $1\leq j\leq n$, such that $t\in w_{jk}'$, where $1\leq k\leq m_j$ and $w_{jk}'\in W_j'$. Therefore, $\pi_A(T)$ is T' with t being removed from w_{jk}'. clearly $REP(\pi_A(T))\subseteq\pi_A(REP(T))$. On the other hand, let r be an arbitrary relation of $REP(T)$, then $r=T_D\cup w_{1i_1}.....w_{ni_n}$, $1\leq i_j\leq m_j$ for $1\leq j\leq n$. If w_{ji_j} is not w_{jk}, then $\pi_A(r)$ is also in $REP(\pi_A(T))$. If w_{ji_j} is w_{jk}, then $\pi_A(r)$ is in $REP(\pi_A(T))$ as well since $t\in w_{jk}'$. Therefore, $\pi_A(REP(T))\subseteq REP(\pi_A(T))$. Thus,

$REPREP, REPREP(REP(\pi_A(T)))=REPREP(\pi_A(REP(T)))$.

Case 4: $\pi_A(T)$ contains the fourth type of redundancies. That is, there exists t of T_D' and W_j', $1\leq j\leq n$, such that $t\in w_{jk}'$, for $k=1,.....,m_j$. The proof for this case is similar to that of Case 3 and is omitted.

Case 5: $\pi_A(T)$ contains the fifth type of redundancies.

(1) if the set of tuple sets to be reduced is not a part of any entanglement, then there exists $W_j'=\{w_{j1}',.....,w_{jm_j}'\}$, $1\leq j\leq n$, such that $w_{jk}'\subseteq w_{jl}'$, where $1\leq k\leq m_j$, $1\leq l\leq m_j$ and $j\neq l$. Thus, $\pi_A(T)$ is T' with w_{jl}' removed from W_j'. Clearly, $REP(\pi_A(T))\subseteq\pi_A(REP(T))$.

Next, we prove that $\pi_A(REP(T))\subseteq REP(\pi_A(T))$. Let $REP(T)=U_1\cup U_2\cup U_3$ such that $U_1\cap U_2\cap U_3=\emptyset$. U_1 is the set of relations which have w_{jk} as the selection from W_j while U_2 is the set of relations which have w_{jl} as the selection from W_j, and U_3 is the set of relations which have neither w_{jk} nor w_{jl} as the selection from W_j. Since $\pi_A(w_{jk})\subseteq\pi_A(w_{jl})$, $U_1\subseteq U_2$. Therefore, $\pi_A(REP(T))=\pi_A(U_1)\cup\pi_A(U_3)$. It is clear from this equation that $\pi_A(REP(T))\subseteq REP(\pi_A(T))$.

(2) the set of tuple sets to be reduced is a part of entanglements: the proof for this case is similar to that of the similar sub-case of Case 1. Therefore, the proof is omitted.

Case 6: T' contains entanglements. The proof for this case is similar to that of Case 1 and is therefore omitted.

Case 7: a set of tuple set of T becomes a singleton set after projection. We assume that, without loss of generality, that $\pi_A(T)$ is reduced. Let $T_E=\{W_1,\ldots,W_k,\ldots,W_n\}$ such that for all $w\in W_k$, $\pi_A(w)=s$. Also let $\pi_A(T)=<T'_D,T'_E>$. Then

$T'_D=T_D\cup s$, and

$T'_E=\{\pi_A(W_1),\ldots,\pi_A(W_{k-1}),\pi_A(W_{k+1}),\ldots,\pi_A(W_n)\}$.

It is obvious that $REP(\pi_A(T))\subseteq\pi_A(REP(T))$. We need to prove that $\pi_A(REP(T))\subseteq REP(\pi_A(T))$. Let r_1 be an arbitrary relation of $REP(T)$ and r_1 has w_{kj}, $1\le j\le m_k$, as the choice from W_k. Then, there exists r_2 such that r_2 is r_1 except it contains w_{kl}, $1\le l\le m_k$ and $l\neq j$, instead of w_{kj}. Note that $\{\pi_{(r_1)}\}_A=\{\pi_A(r_2)\}_A$. Therefore, $\pi_A(r_1)$ is also in $REP(\pi_A(T))$.

Hence, $REPREP(REP(\pi_A(T)))=REPREP(\pi_A(REP(T)))$. □

Theorem 6.4.3.1 $REPREP(REP(T_1\times T_2))=REPREP(REP(T_1)\times REP(T_2))$ for any consistent and reduced E-tables T_1 and T_2.

Proof: Let $T_1=<T^1_D,T^1_E>$, where $T^1_E=\{W^1_1,\ldots,W^1_n\}$ and $T_2=<T^2_D,T^2_E>$, where $T^2_E=\{W^2_1,\ldots,W^2_m\}$, be two arbitrary consistent and reduced E-tables. Let $T=T_1\times T_2$. The only redundancy that could occur in T is the fifth type of redundancy identified in Section 6.3. Let r be an arbitrary relation in $REP(T_1\times T_2)$. According to the definition of $\times$ on Γ and the definition of REP, there exists $r_E\in E$ and $r_F\in F$ such that $r=(T^1_D\times T^2_D)\cup(r_E\times T^2_D)\cup(T^1_D\times r_F)\cup(r_E\times r_F)$. Based on the definition of E and F, it can be derived that there exists $r_1\in REP(T_1)$ and $r_2\in REP(T_2)$ such that $r_E\in r_1$ and $r_F\in r_2$. Further,

$r_1 = T_D^1 \cup r_E$ and $r_2 = T_D^2 \cup r_F$. Therefore, by the definition of $\times$ on Σ, $r_1 \times r_2 \in REP(T_1) \times REP(T_2)$. Since $r_1 \times r_2 = r$, r is also in $(REP(T_1) \times REP(T_2))$.

Now let's prove that $REP(T_1) \times REP(T_2) \subseteq REP(T_1 \times T_2)$. Let r be an arbitrary relation in $REP(T_1) \times REP(T_2)$. Then there exists $r_1 \in REP(T_1)$ and $r_2 \in REP(T_2)$ such that $r = r_1 \times r_2$. According to the definition of REP, $r_1 = T_D^1 \cup \{w_1^1, \ldots, w_n^1\}$, where $w_i^1 \in W_i^1$, $1 \le i \le n$, and $r_2 = T_D^2 \cup \{w_1^2, \ldots, w_m^2\}$, where $w_i^2 \in W_i^2$, $1 \le i \le m$. Therefore, by the definition of $\times$ on Γ, $\{w_1^1, \ldots, w_n^1\} \in E$ and $\{w_1^2, \ldots, w_m^2\} \in F$. Therefore, r is also in $REP(T_1 \times T_2)$.

Hence, $REPREP(REP(T_1 \times T_2)) = REPREP(REP(T_1) \times REP(T_2))$ □

Theorem 6.4.4.1 $REPREP(REP(T_1 - T_2)) = REPREP(REP(T_1) - REP(T_2))$ for any consistent and reduced domain compatible E-tables T_1 and T_2.

Proof: Let $T_1 = <T_D^1, T_E^1>$ with $T_E^1 = \{W_1^1, \ldots, W_n^1\}$ and $T_2 = <T_D^2, T_E^2>$ with $T_E^2 = \{W_1^2, \ldots, W_m^2\}$ be two arbitrary consistent and reduced domain compatible E-tables. Let $T_1 - T_2 = REDUCE(T)$. Note that the redundancies that could arise in T are of the sixth through the eighth types identified in Section 6.3. Let r be an arbitrary relation in $REP(T_1 - T_2)$. According to the definition of $-$ on Γ and the definition of REP, there exists $w_i^1 \in W_i^1$, $1 \le i \le n$, such that

$$r = ((T_D^1 \cup \{w_1^1, \ldots, w_n^1\}) -$$
$$\{t \mid t \in T_D^2 \vee (\exists W)(\exists w)(W \in T_E^2 \wedge w \in W \wedge t \in w)\}) \cup E, \text{ where}$$

E is a set of special dummy values. Note that the portion to the left of $-$ represents a relation of $REP(T_1)$ and the portion to the right of $-$ is equivalent to the union of all the relations of $REP(T_2)$. Therefore, by the definition of $-$ on Σ, $r - E$ is also in $REP(T_1) - REP(T_2)$. By the definition of $REPREP$, $REPREP(T_1 - T_2) \subseteq REPREP(REP(T_1) - REP(T_2))$.

Now let's prove that $REPREP(REP(T_1) - REP(T_2)) \subseteq REPREP(REP(T_1 - T_2))$. Let r be an arbitrary relation in $REP(T_1) - REP(T_2)$. By the definition of $-$ on Σ, $r = r_1 - \bigcup_{r_2 \in U_2} r_2$ for some

r_1 of $REP(T_1)$. This implies that there exists $w_1 \in W_i^1$, $1 \le i \le n$, and $r_1 = T_D^1 \cup \{w_1, \ldots, w_n\}$. By the definition of $-$ on Γ, $T_1 - T_2$ is T_1 with the those tuples that also appear in any of the two components of T_2 replaced by their corresponding dummy values. Therefore, $(r_1 - \lambda) \in REP(T_1 - T_2)$. Hence, $REPREP(REP(T_1) - REP(T_2)) \subseteq REPREP(REP(T_1 - T_2))$.

□

Theorem 6.4.5.1 $REPREP(REP(T_1 \cup T_2)) = REPREP(REP(T_1) \cup REP(T_2))$ for any consistent and reduced domain compatible E-tables T_1 and T_2.

Proof: Let T_1 and T_2 be two arbitrary reduced and consistent domain compatible E-tables. Also let $T_1 \cup T_2 = REDUCE(T)$, assuming that T is consistent. First we note that redundancies could arise in T. Since the union operator preserves dummy values, only the first through the fourth and the ninth types of redundancies identified in Section 6.3 could occur among the members of T_1 and T_2. With these observations in mind, we now prove that $REPREP(REP(T_1) \cup REP(T_2)) = REPREP(REP(T_1 \cup T_2))$.

Let $T_1 = <T_D^1, T_E^1>$, where $T_E^1 = \{W_1^1, \ldots\ldots W_{n_1}^1\}$, and $W_i^1 = \{w_{i1}^1, \ldots\ldots, w_{im_i^1}^1\}$, $1 \le i \le n_1$. Let $T_2 = <T_D^2, T_E^2>$, where $T_E^2 = \{W_1^2, \ldots\ldots W_{n_2}^2\}$, and $W_i^2 = \{w_{i1}^2, \ldots\ldots, w_{im_i^2}^2\}$, $1 \le i \le n_2$. Also let $REP(T_1) \cup REP(T_2) = REDUCE(U)$, assuming that U is consistent.

Case 1: T contains only the first type of redundancies. That is, (a) there exists W_j^1 of T_E^1, $1 \le j \le n_1$, such that $w_{jk}^1 \subseteq T_D^2$, $1 \le k \le m_j^1$, or (b) there exists W_j^2 of T_E^2, $1 \le j \le n_2$, such that $w_{jk}^2 \subseteq T_D^1$, $1 \le k \le m_j^2$. Without loss of generality, assuming that the first situation is the case. There are two possibilities:

(1) none of the tuple sets of $W_j^1 - w_{jk}^1$ entangles the indefinite component of T. Therefore, W_j^1 is to be reduced from T and $T_1 \cup T_2$ is T with W_j^1 removed.

Let r be an arbitrary relation of $REP(T_1 \cup T_2)$. According to the definition of union,

$$r = T_D^1 \cup w_{1l_1^1}^1 \cup \cdots \cup w_{(j-1)l_{(j-1)}^1}^1 \cup w_{(j+1)l_{(j+1)}^1}^1 \cup \cdots \cup w_{n_1 l_{n_1}^1}^1 \cup$$

$T_D^2 \cup w_{1l_1^2}^2 \cup \cdots \cup w_{n_2 l_{n_2}^2}^2$, where

$1 \le l_i^1 \le m_i^1$ for $i = 1, \ldots\ldots, m_1$, and $1 \le l_i^2 \le m_i^2$ for $i = 1, \ldots\ldots, m_m$.

Let $r = r_1 \cup r_2$, where

$r_1 = T_D^1 \cup w_{1l_1^1}^1 \cup \cdots \cup w_{(j-1)l_{(j-1)}^1}^1 \cup w_{(j+1)l_{(j+1)}^1}^1 \cup \cdots \cup w_{n_1 l_{n_1}^1}^1$, where

$1 \le l_i^1 \le m_i^1$ for $i = 1, \ldots\ldots, m_1$, and

$r_2 = T_D^2 \cup w_{1l_1^2}^2 \cup \cdots \cup w_{n_2 l_{n_2}^2}^2$, where $1 \le l_i^2 \le m_i^2$ for $i = 1, \ldots\ldots, m_m$.

Then $r_1 \cup w_{jk}^1$ is in $REP(T_1)$ and r_2 is in $REP(T_2)$. Since $w_{jk}^1 \in T_D^2$, r is also in $REP(T_1) \cup REP(T_2)$.

Conversely, let r be an arbitrary relation of $REP(T_1) \cup REP(T_2)$. Then there exists $r_1 \in REP(T_1)$ and $r_2 \in REP(T_2)$ such that $r = r_1 \cup r_2$. We claim that w_{jk}^1 is in r_1 for otherwise r would be a super set of the relation which contains w_{jk}^1 and therefore does not belong to $REP(T_1) \cup REP(T_2)$ (it should be removed by the *REDUCEACC* operator). Hence,

$r_1 = T_D^1 \cup w_{1l_1^1}^1 \cup \cdots \cup w_{jk}^1 \cup \cdots \cup w_{n_1 l_{n_1}^1}^1$, and

$r_2 = T_D^2 \cup w_{1l_1^2}^2 \cup \cdots \cup w_{n_2 l_{n_2}^2}^2$, where

$1 \le l_i^1 \le m_i^1$ for $i = 1, \ldots\ldots, n_1$ and $1 \le l_i^2 \le m_i^2$ for $i = 1, \ldots\ldots, n_2$.

Then Since $w_{jk}^1 \in T_D^2$, r can be thought of as composed of:

$T_D^1 \cup w_{1l_1^1}^1 \cup \cdots \cup w_{(j-1)l_{(j-1)}^1}^1 \cup w_{(j+1)l_{(j+1)}^1}^1 \cup \cdots \cup w_{n_1 l_{n_1}^1}^1 \cup r_2$, where

$1 \le l_i^1 \le m_i^1$ for $i = 1, \ldots\ldots, n_1$.

Thus, r is also in $REP(T_1 \cup T_2)$.

(2) at least one of the tuple sets of $W_j^1 - w_{jk}^1$ entangles the indefinite component T. For simplicity, we rearrange the indefinite components of $T_1 \cup T_2 = REDUCE(T)$ as follows:

$T = \langle T_D, T_E \rangle$, where

$T_D = T_D^1 \cup T_D^2$, and

$T_E^{'} = \{ W_1^1, W_1^2, W_2^1, \ldots, W_{n_1}^1, W_2^2, \ldots, W_{n_2}^2 \}$, where

1. $W_1^1 = \{ w_{11}^1, w_{12}^1, \ldots, w_{1m_1^1}^1 \}$ such that w_{11}^1 is a subset of T_D^2,
2. Without loss of generality, assuming that W_i^1, $i = 2,\ldots,n_1$, and W_j^2, $j = 2,\ldots,n_2$ are not a part of any entanglements and redundancies (if there were entanglement/redundancies, they are reduced first), and
3. $W_1^2 = \{ w_{11}^2, \ldots, w_{1k}^2, w_{1(k+1)}^2, \ldots, w_{1m_1^2}^2 \}$ such that each of the w_{1i}^2's for $1 \le i \le k$ is free of any elements of $w_{12}^1 \cup \cdots \cup w_{1m_1^1}^1$ and each of the w_{1j}^2 for $(k+1) \le j \le m_1^2$ is a subset of $w_{12}^1 \cup \cdots \cup w_{1m_1^1}^1$.

Hence, according to the definition of union, *REDUCE*, and *REP*, $T^{'} = REDUCE(T) = T_1 \cup T_2$ can be defined as follows:

$T_D^{'} = T_D^1 \cup T_D^2$, and

$T_E = \{ W_2^1, \ldots, W_{n_1}^1, W_2^2, \ldots, W_{n_2}^2, W \}$, where $W = \{ w_{11}^2, \ldots, w_{1k}^2 \}$.

Now let us prove that:

$$REPREP(REP(T_1 \cup T_2)) = REPREP(REP(T_1) \cup REP(T_2)).$$

Let r_1 and r_2 be two arbitrary relations of $REP(T_1)$ and $REP(T_2)$ respectively, there are several possibilities:

1. r_1 contains w_{1i}^1, $2 \le i \le m_1^1$. Then $r_1 \cup r_2$ is a superset of $r_1^{'} \cup r_2$, where $r_1^{'}$ is r_1 with w_{1i}^1, $2 \le i \le m_1^1$, being replaced by w_{11}^1. Therefore, $r_1 \cup r_2$ of this case should be reduced by *REDUCEACC*. Thus, from now on, we assume that there is a relation of $REP(T_1)$ which has w_{11}^1 as its choice for W_1^1 and we denote such an arbitrary valid relation as r_1^1.
2. r_2 contains w, which is one of the tuple sets from $\{ w_{1(k+1)}^2, \ldots, w_{1m_1^2}^2 \}$. There exists a r_1 of $REP(T_1)$ such that r_1 a $w^{'} = w_{1i}^1$, $2 \le i \le m_1^1$, from W_1^1 such that $w^{'} = w$.

Therefore, $r_1^1 \cup r_2 = r_1 \cup r_2$ and consequently should be reduced by *REDUCEACC*.

3. r_2 contains w, which is one of the tuple sets from $\{w_{11}^2, \ldots, w_{1k}^2\}$. Then $r_1^1 \cup r_2$ cannot be reduced for otherwise w would be a part of entanglements or redundancies. Therefore, this is the only case under which $r_1^1 \cup r_2$ is a relation of $REP(T_1) \cup REP(T_2)$.

Therefore, according to the definition of $\cup$ on Γ_R, $REP(T_1) \cup REP(T_2)$ contains the same relations as $REP(T_1 \cup T_2)$.

Note that this proof can easily be generalized to the case where more that two sets of tuple sets are a part of an entanglement.

Case 2: T contains only the second type of redundancies. The proof for this case is similar to Case 1 and is therefore omitted.

Case 3: T contains only the third type of redundancies. That is, (1) there exists a s such that $s \subseteq T_D^1$ and W_j^2 of T_E^2, $1 \le j \le n_2$, such that $s \subset w_{jk}^2$, where $1 \le k \le m_j^2$ and $w_{jk}^2 \in W_j^2$, or (2) there exists a s such that $s \subseteq T_D^2$ and W_j^1 of T_E^1, $1 \le j \le n_1$, such that $s \subset w_{jk}^1$, where $1 \le k \le m_j$ and $w_{jk}^1 \in W_j^1$. It is straightforward that $REP(T_1 \cup T_2) = REP(T_1) \cup REP(T_2)$.

Case 4: T contains only the fourth type of redundancies. That is, (1) there exists s such that $s \subseteq T_D^1$ and W_j^2 of T_E^2, $1 \le j \le n_2$, such that $s \subset w_{jk}^2$ for $k = 1, \ldots, m_j^2$, where $w_{jk}^2 \in W_j^2$, $1 \le k \le m_j^2$, or (2) there exists s such that $s \subseteq T_D^2$ and W_j^1 of T_E^1, $1 \le j \le n_1$, such that $s \subset w_{jk}^1$ for $k = 1, \ldots, m_j^1$, where $w_{jk}^1 \in W_j^1$, $1 \le k \le m_j^1$. It is straightforward that $REP(T_1 \cup T_2) = REP(T_1) \cup REP(T_2)$.

Case 5: T contains entanglement. The proof for this case is similar to that of the second situation of Case 1. Therefore, the proof for this case is omitted.

Hence, $REPREP(REP(T_1 \cup T_2)) = REPREP(REP(T_1) \cup REP(T_2))$. □

Theorem 6.4.6.1 $REPREP(REP(T_1 \cap T_2)) = REPREP(REP(T_1) \cap REP(T_2))$ for any consistent and reduced domain compatible E-tables T_1 and T_2.

Proof: Let T_1 and T_2 be two arbitrary reduced and consistent domain compatible E-tables. Let $T=T_1 \cap T_2$. We make the following observations first based on the assumption that T is consistent: (1) T is reduced, and (2) T could contain entanglements (because of special dummy values). for the proof follows, we also make the following convention: let W be a set of tuple sets and S be a set, then $W \subseteq S$ if $w \subseteq S$ for every tuple set w of W, where $w \subseteq S$ is defined as: $w \subseteq S$ if $t \in S$ for every $t \in w$ and t is not a dummy value. With the above observations and convention in mind, we now prove that $REPREP(REP(T_1) \cap REP(T_2)) = REPREP(REP(T_1 \cap T_2))$.

Let $T_1 = <T_D^1, T_E^1>$, where

$T_E^1 = \{ W_1^1, \ldots, W_k^1, W_{k+1}^1, \ldots, W_{n_1}^1 \}$, and $W_i^1 = \{ w_{i1}^1, \ldots\ldots, w_{im_i^1}^1 \}$, $1 \le i \le n_1$ such that:

(1) $(\forall i)(1 \le i \le k \rightarrow W_i^1 \subseteq T_D^2)$, and

(2) $(\forall i)((k+1) \le i \le n_1 \rightarrow \neg(W_i^1 \subseteq T_D^2))$.

Also let $T_2 = <T_D^2, T_E^2>$, where

$T_E^2 = \{ W_1^2, \ldots, W_l^2, W_{l+1}^2, \ldots, W_{n_2}^2 \}$, and $W_i^2 = \{ w_{i1}^2, \ldots\ldots, w_{im_i^2}^2 \}$, $1 \le i \le n_2$ such that:

(3) $(\forall i)(1 \le i \le l \rightarrow W_i^2 \subset T_D^1)$, and

(4) $(\forall i)((l+1) \le i \le n_2 \rightarrow \neg(W_i^2 \subseteq T_D^1))$.

Finally let $REP(T_1) \cap REP(T_2) = U$, assuming that U is consistent.

Let r be an arbitrary relation of $REP(T_1 \cap T_2)$. There are two possibilities:

(1) T is reduced: in this case, according to (1), (2), (3), and (4), we have:

$r = (T_D^1 \cap T_D^2) \cup w_{1j_1^1}^1, \ldots, w_{kj_k^1}^1 \cup w_{1j_1^2}^2 \cup \cdots \cup w_{lj_l^2}^2$, where

$1 \le j_i^1 \le m_i^1$.

It should be clear that there exists $r_1 \in REP(T_1)$ and $r_2 \in REP(T_2)$ such that:

$r_1 = T_D^1 \cup w_{1j_1^1}^1, \ldots, w_{kj_k^1}^1 \cup w_{(k+1)j_{k+1}^1}^1 \cup \cdots \cup w_{n_1}^1$, and

$r_2 = T_D^1 \cup w_{1j_1^2}^2, \ldots, w_{lj_l^2}^2 \cup w_{(l+1)j_{l+1}^2}^2 \cup \cdots \cup w_{n_2}^2$.

Therefore, $r_1 \cap r_2 = r$ and is in $REP(T_1) \cap REP(T_2)$.

(2) T contains entanglements: note that in this case, the entanglement happens because of special dummy values. Suppose that there exists a i, $1 \le i \le k$, and a j, $1 \le j \le l$, such that W_i^1 and W_j^2 form the entanglement in T. Then W_i^1 is of the form:

$\{\varepsilon, w_{i1}^1, \ldots, w_{im_i^1}^1\}$, and

W_j^2 is of the form:

$\{\varepsilon, w_{j1}^2, \ldots, w_{jm_j^2}^2\}$, and

Therefore, $T_1 \cap T_2 = REDUCE(T) = T' = \langle T_D', T_E' \rangle$ is of the form:

$T_D' = (T_D^1 \cap T_D^2)$

$T_E' = \{W_1^1, \ldots, W_{i-1}^1, W_{i+1}^1, \ldots, W_k^1, W_1^2, \ldots, W_{j-1}^2, W_{j+1}^2, \ldots, W_l^2,$

$\{\{\varepsilon\}, \{w_{i1}^1 \cup w_{j1}^2\}, \ldots, \{w_{i1}^1 \cup w_{jm_j^2}^2\}, \{w_{im_i^1}^1 \cup w_{j1}^2\}, \ldots, \{w_{im_i^1}^1 \cup w_{jm_j^2}^2\}\}\}$.

Let r be an arbitrary relation of $REP(T_1 \cap T_2)$, there are then two cases:

Case 1:

$r = (T_D^1 \cap T_D^2) \cup w_{1g_1^1}^1 \cup \ldots \cup w_{(i-1)g_{i-1}^1}^1 \cup w_{(i+1)g_{i+1}^1}^1 \cup \cdots \cup w_{kg_k^1}^1 \cup$

$w_{1g_1^2}^2 \cup \ldots \cup w_{(j-1)g_{j-1}^2}^2 \cup w_{(j+1)g_{j+1}^2}^2 \cup \cdots \cup w_{lg_l^2}^2 \cup \varepsilon$, where

$1 \le g_p^1 \le m_p^1$, $1 \le p \le k$, and $1 \le g_q^2 \le m_q^2$, $1 \le q \le l$.

It should be clear that there exists $r_1 \in REP(T_1)$ such that:

$r_1 = T_D^1 \cup w_{1g_1^1}^1 \cup \ldots \cup w_{(i-1)g_{i-1}^1}^1 \cup \varepsilon \cup$

$w_{(i+1)g_{i+1}^1}^1 \cup \cdots \cup w_{kg_k^1}^1 \cdots \cup w_{m_1 g_{m_1}^1}^1$, where

$1 \le g_p^1 \le m_p^1$, $1 \le p \le n_1$, and

There exists $r_2 \in REP(T_2)$ such that:

$r_2 = T_D^2 \cup w_{1g_1^2}^2 \cup \ldots \cup w_{(j-1)g_{j-1}^2}^2 \cup \varepsilon \cup$

$w_{(j+1)g_{j+1}^2}^2 \cup \cdots \cup w_{lg_l^2}^2 \cdots \cup w_{m_2 g_{m_2}^2}^2$, where

$1 \le g_q^1 \le m_q^2$, $1 \le q \le n_2$.

It should be clear that $r_1 \cap r_2 = r - \lambda$ and $r_1 \cap r_2$ is in $REP(T_1) \cap REP(T_2)$. Therefore, $REPREP(REP(T_1 \cap T_2)) \subseteq REPREP(REP(T_1) \cap REP(T_2))$.

Case 2:

$$r=(T_D^1\cap T_D^2)\cup w^1_{1g_1^1}\cup\ldots\cup w^1_{(i-1)g^1_{i-1}}\cup w^1_{(i+1)g^1_{i+1}}\cup\cdots\cup w^1_{kg_k^1}\cup$$

$$w^2_{1g_1^2}\cup\ldots\cup w^2_{(j-1)g^2_{j-1}}\cup w^2_{(j+1)g^2_{j+1}}\cup\cdots\cup w^2_{lg_l^2}\cup$$

$(w^1_{ip}\cup w^2_{jq})$, where

$1\le g_e^1\le m_e^1$, $1\le e\le k$, and $1\le g_f^2\le m_f^2$, $1\le f\le l$. Further, $1\le p\le m_i^1$ and $1\le q\le m_j^2$.

Then there exists $r_1\in REP(T_1)$ such that:

$$r_1=T_D^1\cup w^1_{1g_1^1}\cup\ldots\cup w^1_{(i-1)g^1_{i-1}}\cup w^1_{ip}\cup w^1_{(i+1)g^1_{i+1}}\cup\cdots\cup w^1_{kg_k^1}\cdots\cup w^1_{m_1g_{m_1}{}^1},$$

where

$1\le g_p^1\le m_p^1$, $1\le p\le n_1$, and $1\le p\le m_i^1$, and

There exists $r_2\in REP(T_2)$ such that:

$$r_2=T_D^2\cup w^2_{1g_1^2}\cup\ldots\cup w^2_{(j-1)g^2_{j-1}}\cup w^2_{jq}\cup w^2_{(j+1)g^2_{j+1}}\cup\cdots\cup w^2_{lg_l^2}\cdots\cup w^2_{m_2g_{m_2}{}^2},$$

where

$1\le g_q^1\le m_q^2$, $1\le q\le n_2$ and $1\le q\le m_j^2$.

Similar to Case 1, $r_1\cap r_2=r-\lambda$ and $r_1\cap r_2$ is in $REP(T_1)\cap REP(T_2)$. Therefore,

$REPREP(REP(T_1\cap T_2))\subseteq REPREP(REP(T_1)\cap REP(T_2))$.

By applying the same reasoning, it can be shown that

$REPREP(REP(T_1)\cap REP(T_2))\subseteq REPREP(REP(T_1\cap T_2))$. □